View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

European Journal of Combinatorics 33 (2012) 1330–1368

Contents lists available at SciVerse ScienceDirect

European Journal of Combinatorics

journal homepage: www.elsevier.com/locate/ejc

Groups, graphs, languages, automata, games and second-order monadic logic

Tullio Ceccherini-Silberstein [a], Michel Coornaert [b], Francesca Fiorenzi [c], Paul E. Schupp [d]

[a] *Dipartimento di Ingegneria, Università del Sannio, C.so Garibaldi 107, 82100 Benevento, Italy*

[b] *Institut de Recherche Mathématique Avancée, UMR 7501, Université de Strasbourg et CNRS, 7 rue René-Descartes, 67000 Strasbourg, France*

[c] *Laboratoire de Recherche en Informatique, Université Paris-Sud 11, 91405 Orsay Cedex, France*

[d] *Department of Mathematics, University of Illinois at Urbana–Champaign, 1409 W. Green Street (MC-382) Urbana, IL 61801-2975, USA*

ARTICLE INFO

Article history:
Available online 12 April 2012

Dedicated to Toni Machì on the occasion of his 70th birthday.

ABSTRACT

In this paper we survey some surprising connections between group theory, the theory of automata and formal languages, the theory of ends, infinite games of perfect information, and monadic second-order logic.

© 2012 Published by Elsevier Ltd

1. Introduction

In this survey we discuss some interleaved strands of ideas connecting the items in the title. We do not, of course, develop all the connections between groups and automata. In particular, we do not consider either *automatic groups* (see, for instance, the monograph [33] by Epstein, Cannon, Hold, Levy, Paterson, and Thurston) or *automata groups*, also called *self-similar groups* (including the well known Grigorchuk group of intermediate growth [39,28]: see, for instance, [41,5,6] and the monograph [82] by Nekrashevych).

A finitely generated group can be described by a presentation $G = \langle X; R \rangle$ in terms of generators and defining relators. In this case, the *group alphabet* is $\Sigma = X \cup X^{-1}$. Anisimov [2] introduced the fruitful point of view of considering the Word Problem of $G = \langle X; R \rangle$ as the formal language $\mathrm{WP}(G : X; R) = \{w \in \Sigma^* : w = 1_G\}$. Although the Word Problem is generally a very complicated set, Anisimov asked what one could say about the group G if $\mathrm{WP}(G : X; R)$ is a regular or context-free language in the usual sense of formal language theory. He showed that a finitely generated group has

E-mail addresses: tceccher@mat.uniroma3.it (T. Ceccherini-Silberstein), coornaert@math.unistra.fr (M. Coornaert), fiorenzi@lri.fr (F. Fiorenzi), schupp@math.uiuc.edu (P.E. Schupp).

0195-6698/$ – see front matter © 2012 Published by Elsevier Ltd
doi:10.1016/j.ejc.2012.03.010

regular Word Problem if and only if the group is finite. An important class of groups is the class of virtually free groups, that is, groups having a free subgroup of finite index. Muller and Schupp [76] showed that a finitely generated group has context-free Word Problem if and only if the group is virtually free.

The basic geometric object associated with a finitely generated group $G = \langle X; R\rangle$, its *Cayley graph* $\Gamma(G : X; R)$, was already defined by Cayley [13] in 1878. Intuitively, an *end* (a notion due to Hopf [50] and Freudenthal [34]) of a locally finite graph is a way to go to infinity in the graph. The number of ends of a connected graph Γ with origin v_0 is the limit, as n goes to infinity, of the number of infinite connected components of $\Gamma \setminus \Gamma_n$, where the n-ball Γ_n consists of all vertices and edges on paths of length less than or equal to n starting at v_0. The number of ends of a finitely generated group is the number of ends of its Cayley graph. (It is not obvious, but true, that this number depends only on the group and not on the particular presentation chosen.) The proof of the characterization of groups with context-free Word Problem depends heavily on the Stallings structure theorem [92], which shows that finitely generated groups with more than one end must have a particular algebraic structure.

It turns out that the connection between ends and context-freeness is much deeper than just the case of groups. It is well-known [22,46,49] that a formal language is context-free if and only if it is the language accepted by some pushdown automaton. The concept of a *finitely generated graph* gives a common framework in which one can discuss both Cayley graphs of finitely generated groups and complete transition graphs of various kinds of automata, in particular the complete transition graph of a pushdown automaton.

Instead of considering the number of ends of a finitely generated graph Γ, one can consider the number $c(\Gamma)$ of labelled graph isomorphism classes of connected components of $\Gamma \setminus \Gamma_n$ over *all* components and all $n \geq 1$. Say that Γ has *finitary end-structure* if $c(\Gamma) < \infty$. Muller and Schupp [77] proved that a finitely generated graph has finitary end-structure if and only if Γ is isomorphic to the complete transition graph $\Gamma(M)$ of some pushdown automaton M.

One of the most powerful positive results about decision problems in logic is Rabin's theorem [85] that the second-order monadic theory of the rooted infinite binary tree T_2 is decidable. This theory, $S2S$, is the *theory of two successor functions*, as we now explain. We consider the infinite binary tree as the rooted tree with root v_0 and right successor edges labelled by 1 and left successor edges labelled by 0. The second-order monadic logic of T_2 has variables ranging over arbitrary sets of vertices. We have two set-valued successor functions: if S is a set of vertices and $a \in \{0, 1\}$ then $Sa = \{va : v \in S\}$. There is also the relation symbol $\subseteq$ for set inclusion and a constant symbol v_0 for the origin. There are the usual quantifiers $\forall$, $\exists$ and the Boolean connectives $\wedge$ (and), $\vee$ (or), and $\neg$ (negation). Some formulations include individual variables for single vertices, but sets with a single element are definable, as is equality. The great power of this language is that one can quantify over arbitrary sets of vertices.

The characterization of graphs with finitary end structure shows that such graphs are "very tree-like". Indeed, such a graph Γ contains a regular subtree of finite index, in the sense that there is a subtree T defined by a finite automaton and a fixed bound $D \geq 0$ such that every vertex in Γ is within distance D of some vertex in the subtree T. From this fact, it is possible to reduce questions about the monadic theory of Γ to questions about the monadic theory of the tree T. It then follows from Rabin's theorem that the monadic theory of the complete transition graph of any pushdown automaton is decidable. In particular, if $G = \langle X; R\rangle$ is any finitely generated presentation of a virtually free finitely generated group then the monadic second-order theory of its Cayley graph $\Gamma(G : X; R)$ is decidable. There are finitely generated graphs which do not have finitary end structure but whose monadic theories are decidable. However, Kuske and Lohrey [59] have recently proved that if the monadic theory of the Cayley graph of a finitely generated group is decidable then the group must be virtually free.

There is an interesting application of the decidability of the monadic second-order theory of Cayley graphs of context-free groups to the theory of cellular automata on groups. The following definition is actually a straightforward generalization of von Neumann's concept [94] of cellular automata on the grid on integer lattice points in the plane, that is, the Cayley graph of $\mathbb{Z}^2$. Let G be a group and Σ a finite set and denote by Σ^G the set of all maps $\alpha\colon G \to \Sigma$. Equip Σ^G with the action of G defined by

$$g(\alpha)(h) = \alpha(g^{-1}h) \quad \text{for all } \alpha \in \Sigma^G \text{ and } g, h \in G.$$

Then one says that a map $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is a *cellular automaton* provided there exists a *finite* subset $M \subset G$ and a map $\mu\colon \Sigma^M \to \Sigma$ such that

$$\mathcal{C}(\alpha)(g) = \mu((g^{-1}\alpha)|_M) \tag{1.1}$$

for all $\alpha \in \Sigma^G$ and $g \in G$, and where $(\cdot)|_M$ denotes the restriction to M. One is often interested in determining whether or not a cellular automaton is surjective (respectively, injective, bijective). In particular, the following decision problem naturally arises: given a finite subset $M \subset G$ and a map $\mu\colon \Sigma^M \to \Sigma$, is the associated cellular automaton $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ defined in (1.1) surjective (respectively injective, bijective) or not? Amoroso and Patt [1] proved in 1972 that if $G = \mathbb{Z}$ the above problem is decidable. If follows from the decidability of the monadic second-order theory of Cayley graphs of context-free groups that the problem for cellular automata defined over virtually-free groups is decidable. On the other hand, Kari [52–54] proved that if $G = \mathbb{Z}^d$, $d \geq 2$, this problem is undecidable. His proof is based on Berger's [7] undecidability result for the *Domino Problem for Wang tiles*.

In 1960 Büchi [11] proved that the monadic theory of $\mathbb{N}$ with one successor function, $S1S$, is decidable by introducing finite automata working on infinite words. Monadic sentences are too complicated to deal with directly and the idea is to effectively associate with each monadic sentence ϕ a finite automaton $\mathcal{A}_\phi$ such that ϕ is true if and only if the language $L(\mathcal{A}_\phi)$ accepted by $\mathcal{A}_\phi$ is nonempty. Of course, one must carefully define what it means for an automaton to accept an infinite word. Rabin used automata working on infinite trees to establish a similar correspondence between sentences of $S2S$ and the Emptiness Problem for tree automata.

The theory of automata working on infinite inputs is thus crucial to studying monadic theories, but proving theorems about such automata is difficult. The best way to understand such automata is in terms of infinite games of perfect information as introduced by Gale and Stewart [36]. Let Σ be a finite alphabet and let $\Sigma^{\mathbb{N}}$ denote the set of all infinite words $w = a_1a_2\cdots a_n\cdots$ over Σ (all the infinite words which we consider are infinite to the right). Let $\mathcal{W}$ be a subset of $\Sigma^{\mathbb{N}}$. We consider the following game between Player I and Player II: Player I chooses a letter $\sigma_1 \in \Sigma$ and Player II then chooses a letter $\sigma_2 \in \Sigma$. Continuing indefinitely, at step n Player I chooses a letter $\sigma_{2n-1} \in \Sigma$ and Player II then chooses a letter $\sigma_{2n} \in \Sigma$. This sequence of choices defines an infinite word $w \in \Sigma^{\mathbb{N}}$. Player I wins the game if $w \in \mathcal{W}$ and Player II wins otherwise. The basic question about such games is whether or not one of the players has a winning strategy, that is, a map $\phi : \Sigma^* \to \Sigma$ such that when a finite word u has already been played, the player using the strategy then plays $\phi(u) \in \Sigma$ and always wins. Using the Axiom of Choice, it is possible to construct winning sets such that neither player has a winning strategy, but this cannot happen if the set $\mathcal{W}$ is not "too complicated". An important theorem of Martin [68,69] shows that if the set $\mathcal{W}$ is a Borel set then one of the two players must have a winning strategy.

To apply infinite games to automata, given an automaton M one defines the *acceptance game* $\mathcal{G}(M, w)$ for M on an infinite input $w \in \Sigma^{\mathbb{N}}$. The first player wins if M accepts w while the second player wins if M rejects. In the case of automata, the winning condition of the acceptance game is at the second level of the Borel hierarchy so one of the players has a winning strategy. This essentially proves closure under complementation of regular languages in $\Sigma^{\mathbb{N}}$. The situation is similar for automata on the binary tree. The celebrated "Forgetful Determinacy Theorem" of Gurevich and Harrington [44] states that a fixed finite amount of memory, *the later appearance record*, is all that a winning strategy needs to take into account.

The paper is organized as follows. In Section 2 we review the notions of regular, context-free, and computably enumerable languages together with the parallel notions of grammars and their associated classes of automata: finite-state automata, pushdown automata, and Turing machines. Section 3 is devoted to presentations of finitely generated groups and their associated Cayley graphs. We consider the Word Problem for a finitely generated group as a formal language. We prove Anisimov's characterization of groups with regular Word Problem and present the Muller–Schupp characterization of groups with context-free Word Problem. We also discuss some applications of formal language theory to subgroups and present Haring-Smith's characterization of basic groups in terms of their Word Problem. In Section 4 we consider the notion of a finitely generated graph and the number of ends of a finitely generated graphs together with Stallings Structure Theorem and the notion of accessibility. We then consider the notion of finitely generated graphs with finitary end-structure and their characterization as complete transition graphs of pushdown automata. Section 5 is devoted to second-order monadic logic where we discuss Büchi's theorem on the decidability of second-order monadic theory $S1S$ and Rabin's theorem on the decidability of second-order monadic theory $S2S$ of the infinite binary tree. We then discuss the decidability of second-order monadic theory for complete transition graphs of pushdown automata. We consider the classical Domino Problem and its undecidability due to Berger and Robinson. After generalizing the Domino Problem to finitely generated groups, we show that it is decidable for virtually free groups. In Section 6 we consider the Surjectivity, Injectivity, and Bijectivity problems for cellular automata on finitely generated groups and its decidability for virtually free groups. The last section is devoted to finite automata on infinite inputs and the work of Büchi, of Rabin, and of Muller and Schupp. We then discuss infinite games of perfect information, the theorems of Davis and Martin, and the Forgetful Determinacy Theorem of Gurevich and Harrington.

2. Languages, grammars, and automata

2.1. The free monoid over a finite alphabet

Let Σ be a finite *alphabet*, that is, a finite set of *letters*. A *word* on Σ is any element of the set

$$\Sigma^* = \bigcup_{n=0}^{\infty} \Sigma^n,$$

where $\Sigma^n = \{a_1 a_2 \cdots a_n : a_k \in \Sigma, 1 \le k \le n\}$. The number $|w| = n$ is the *length* of the word $w = a_1 a_2 \cdots a_n$. The unique word of length zero is denoted by ε and is called the *empty word*.

The *concatenation* of two words $w = a_1 a_2 \cdots a_n \in \Sigma^n$ and $w' = a'_1 a'_2 \cdots a'_m \in \Sigma^m$ is the word $ww' \in \Sigma^{n+m}$ defined by

$$ww' = a_1 a_2 \cdots a_n a'_1 a'_2 \cdots a'_m. \tag{2.1}$$

We have $\varepsilon w = w \varepsilon = w$ and $(ww')w'' = w(w'w'')$ for all $w, w', w'' \in \Sigma^*$. Thus, Σ^* is a monoid under the concatenation product with identity element the empty word ε. The monoid Σ^* satisfies the following universal mapping property: if M is any monoid, then every map $f : \Sigma \to M$ uniquely extends to a monoid homomorphism $\varphi : \Sigma^* \to M$. Due to this property, Σ^* is the *free monoid* over Σ.

Let u, w be two words over Σ. One says that u is a *subword* of w if there exist $u_1, u_2 \in \Sigma^*$ such that $w = u_1 u u_2$.

A *language* over Σ is a subset $L \subset \Sigma^*$.

2.2. Context-free languages

In this section, we discuss the class of context-free languages introduced by Chomsky [23].

A *context-free grammar* is a quadruple $\mathcal{G} = (V, \Sigma, P, S_0)$, where V is a finite set of *variables*, disjoint from the finite alphabet Σ of *terminal symbols*. The variable $S_0 \in V$ is the *start symbol*, and $P \subset V \times (V \cup \Sigma)^*$ is a finite set of *production rules*. We write $S \vdash u$ if $(S, u) \in P$. For $v, w \in (V \cup \Sigma)^*$, we write $v \Longrightarrow w$ if $v = v_1 S v_2$ and $w = v_1 u v_2$, where $u, v_1, v_2 \in (V \cup \Sigma)^*$ and $S \vdash u$. The expression $v \Longrightarrow w$ is a single *derivation step*, and it is called *rightmost* if $v_2 \in \Sigma^*$. A *derivation* is a sequence $v = w_0, w_1, \ldots, w_n = w \in (V \cup \Sigma)^*$ such that $w_i \Longrightarrow w_{i+1}$ for each $i = 0, \ldots, n-1$ and we then write $v \overset{*}{\Longrightarrow} w$. A *rightmost derivation* is one where each step is rightmost. It can be easily shown that if $v \overset{*}{\Longrightarrow} w$ with $w \in \Sigma^*$, then there exists a rightmost derivation $v \overset{*}{\Longrightarrow} w$. For $S \in V$, we consider the language $L_S = \{w \in \Sigma^* : S \overset{*}{\Longrightarrow} w\}$. The *language generated by* $\mathcal{G}$ is

$$L(\mathcal{G}) := L_{S_0} = \{w \in \Sigma^* : S_0 \overset{*}{\Longrightarrow} w\}.$$

A *context-free language* is a language generated by a context-free grammar.

Example 2.1 (*The Dyck Language*)**.** The language of all correctly balanced expressions involving several types of parentheses is in some sense the "primordial" context-free language. Let $n \geq 1$ and $\Sigma = \{a_1, \bar{a}_1, \ldots, a_n, \bar{a}_n\}$. Consider the grammar $\mathcal{G}$ with one single variable S_0 and productions $S_0 \vdash \varepsilon$ and $S_0 \vdash a_i S_0 \bar{a}_i S_0$, $i = 1, \ldots, n$. The language $L(\mathcal{G})$ generated by the grammar $\mathcal{G}$ is called the *Dyck language*. Thinking of the a_i's (resp. $\bar{a}_i$'s) as n different "open" (resp. "closed") parenthesis symbols, then $L(\mathcal{G})$ consists of all correctly nested parenthesis expressions over these symbols. For example,

$$\begin{aligned} S_0 &\vdash a_2S_0\bar{a}_2S_0 \Longrightarrow a_2S_0\bar{a}_2a_1S_0\bar{a}_1S_0 \Longrightarrow a_2S_0\bar{a}_2a_1S_0\bar{a}_1 \\ &\Longrightarrow a_2S_0\bar{a}_2a_1a_2S_0\bar{a}_2S_0\bar{a}_1 \Longrightarrow a_2S_0\bar{a}_2a_1a_2S_0\bar{a}_2\bar{a}_1 \Longrightarrow a_2S_0\bar{a}_2a_1a_2\bar{a}_2\bar{a}_1 \\ &\Longrightarrow a_2\bar{a}_2a_1a_2\bar{a}_2\bar{a}_1 \end{aligned}$$

is the unique rightmost derivation of $a_2\bar{a}_2a_1a_2\bar{a}_2\bar{a}_1 \in L(\mathcal{G})$.

A context-free grammar $\mathcal{G} = (V, \Sigma, P, S_0)$ and its associated language $L(\mathcal{G})$ are called *linear* if every production rule in P is of the form $S \vdash v_1Tv_2$ or $S \vdash v$, where $v, v_1, v_2 \in \Sigma^*$ and $S, T \in V$. If in this situation one always has $v_2 = \varepsilon$ (the empty word), then the grammar and language are called *right linear*. Similarly, the grammar and language are *left linear* if one always has $v_1 = \varepsilon$. It is well known (cf. [22,46,49]) that both left linear and right linear grammars generate the same class of languages, namely, the class of *regular* languages.

Example 2.2 (*Palindromes*)**.** Let Σ be a finite alphabet. A word $w = a_1a_2\cdots a_n$ is a *palindrome* provided that $a_i = a_{n-i+1}$ for all $i = 1, 2, \ldots, n$, that is, w is the same read both forwards and backwards. We denote by $L_{\mathrm{pal}}(\Sigma)$ the language consisting of all palindromes over the alphabet Σ. For example, $L_{\mathrm{pal}}(\{a\}) = \{a\}^* = \{\varepsilon, a, aa, aaa, \ldots\}$ and

$$L_{\mathrm{pal}}(\{a, b\}) = \{\varepsilon, a, b, aa, bb, aaa, aba, bab, bbb, aaaa, abba, baab, bbbb, \ldots\}.$$

Consider the grammar $\mathcal{G}$ with a unique variable S_0 and productions of the form $S_0 \vdash \varepsilon$, $S_0 \vdash a$ and $S_0 \vdash aS_0a$, for each $a \in \Sigma$. Then $\mathcal{G}$ is a linear grammar and $L(\mathcal{G}) = L_{\mathrm{pal}}(\Sigma)$. It follows that the language consisting of all palindromes is linear.

Example 2.3 (*The Free Group*)**.** Let X be a finite set and denote by F_X the free group based on X. (If n denotes the cardinality of X we shall also denote F_X by F_n and refer to it as to the *free group of rank* n.) Let X^{-1} be a disjoint copy of X and set $\Sigma = X \cup X^{-1}$. We denote by $x \mapsto x^{-1}$ the involutive map on Σ exchanging X and X^{-1} so $(x^{-1})^{-1} = x$ for all $x \in X$. A word $w \in \Sigma^*$ is *reduced* if it contains no subword of the form xx^{-1} or $x^{-1}x$ for $x \in X$. For example, if $x, y \in X$ are distinct, then the words $\varepsilon, x, xy, xy^{-1}, xy^{-1}x^{-1}$ are reduced, while $xx^{-1}, x^{-1}xy$ are not. We denote by $L_{\mathrm{red}}(\Sigma) \subset \Sigma^*$ the language consisting of all reduced words. It is well known that every element of F_X has a unique representative as a reduced word in $L_{\mathrm{red}}(\Sigma)$.

Consider the grammar $\mathcal{G} = (V, \Sigma, P, S_0)$ where $V = \{S_0\} \cup \{S_x : x \in \Sigma\}$ and P consists of the productions of the form

$$S_0 \vdash \varepsilon \quad \text{and} \quad S_0 \vdash xS_x \quad \text{for all } x \in \Sigma$$

and

$$S_x \vdash \varepsilon \quad \text{and} \quad S_x \vdash yS_y \quad \text{for all } y \in \Sigma \setminus \{x^{-1}\}$$

for all $x \in \Sigma$. Note that $\mathcal{G}$ is a right-linear grammar and that $L(\mathcal{G}) = L_{\mathrm{red}}(\Sigma)$. Thus, the language of all reduced words over Σ is regular.

Returning to a general context-free grammar $\mathcal{G}$, for a given variable $S \in V$, we define the *degree of ambiguity*, $d_S(w)$, of a word $w \in \Sigma^*$ as the number of different rightmost derivations $S \overset{*}{\Longrightarrow} w$. We have $d_S(w) > 0$ if and only if $w \in L_S$. The grammar is called *unambiguous* if $d_{S_0}(w) = 1$ for all $w \in L(\mathcal{G})$. Otherwise, if there exists $w \in L(\mathcal{G})$ such that $d_{S_0}(w) > 1$, the grammar is called *ambiguous*. A context-free language L is called *unambiguous* if it is generated by some unambiguous grammar and *inherently ambiguous* if all context-free grammars generating L are ambiguous. It is a fact that there exist inherently ambiguous context-free languages (cf. [49]).

2.3. Growth of context-free languages

Let Σ be a finite alphabet and $L \subset \Sigma^*$ a language.
The *growth function* of L is the map $\gamma_L : \mathbb{N} \to \mathbb{N}$ defined by

$$\gamma_L(n) = |\{w \in L : |w| \le n\}|, \quad n \in \mathbb{N}.$$

Note that

$$\gamma_L(n) \le \gamma_{\Sigma^*}(n) = \sum_{k=0}^{n} |\Sigma|^k = \frac{|\Sigma|^{n+1} - 1}{|\Sigma| - 1} \le |\Sigma|^{n+1} = C|\Sigma|^n$$

for all $n \in \mathbb{N}$ where $C = |\Sigma|$. It follows that there exist $C > 0$ and $a > 1$ such that

$$\gamma_L(n) \le Ca^n \tag{2.2}$$

for all $n \in \mathbb{N}$.

The *growth rate* of L is the number

$$\lambda(L) = \limsup_{n\to\infty} \gamma_L(n)^{\frac{1}{n}}. \tag{2.3}$$

On says that L is of *exponential growth* if $\lambda(L) > 1$. Otherwise, if $\lambda(L) = 1$, then L is of *sub-exponential growth*. Note that L is of exponential growth if and only if there exists $a > 1$ such that $\gamma_L(n) \ge a^n$ for all $n \in \mathbb{N}$. A language L is said to be of *polynomial growth* provided that there exist an integer $d \ge 0$ and a constant $C > 0$ such that $\gamma_L(n) \le C + Cn^d$ for all $n \in \mathbb{N}$. Finally, one says that L is of *intermediate growth* if its growth is sub-exponential but not polynomial. Note that a language cannot be of "super-exponential growth" by virtue of (2.2).

Bridson and Gilman [9] and, independently, Incitti [51], proved that the growth of a context-free language is either polynomial or exponential. An explicit algorithm for determining this alternative is presented in [14]. On the other hand, Grigorchuk and Machì [40] presented an example of an *indexed language* of intermediate growth. (The class of indexed languages, introduced by Aho, properly contains the class of context-free languages and, in turn, is properly contained in the class of computably enumerable languages.)

One says that the language L is *growth-sensitive* if

$$\lambda(L^F) < \lambda(L)$$

for every non-empty $F \subset \Sigma^*$ consisting of subwords of elements of L, where

$$L^F = \{w \in L : \text{no } v \in F \text{ is a subword of } w\}.$$

It is a well known fact, which can be deduced from the Perron–Frobenius theory (see [17] for an alternative proof), that regular languages are growth-sensitive. Ceccherini-Silberstein and Woess [19,15] (see also [20]) extended this result to all unambiguous ergodic context-free languages. (Here "ergodicity" corresponds to strong connectedness of the dependency graph (in the sense of Kuich [58]) associated with an unambiguous context-free grammar generating the language.)

2.4. Finite automata

A *nondeterministic finite automaton* is a 5-tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where Q is a nonempty finite set of *states*, Σ is a finite alphabet, $q_0 \in Q$ is the *initial state*, $F \subset Q$ is the set of *final states*, and the map

$$\delta : Q \times \Sigma \to \mathcal{P}(Q)$$

is the *transition function*. (As usual, $\mathcal{P}(Q)$ denotes the set of all subsets of Q.) The automaton works as follows. When reading a word $w \in \Sigma^*$, letter by letter, from left to right, it can change its state according to the transition function. A *run* of $\mathcal{A}$ on a word $w = \sigma_1\sigma_2\cdots\sigma_n$ is a function $\rho : \{0, 1, \ldots, n+1\} \to Q$ such that $\rho(0) = q_0$ and $\rho(i+1) \in \delta(\rho(i), \sigma_i)$ for $i = 0, 1, \ldots, n$. A

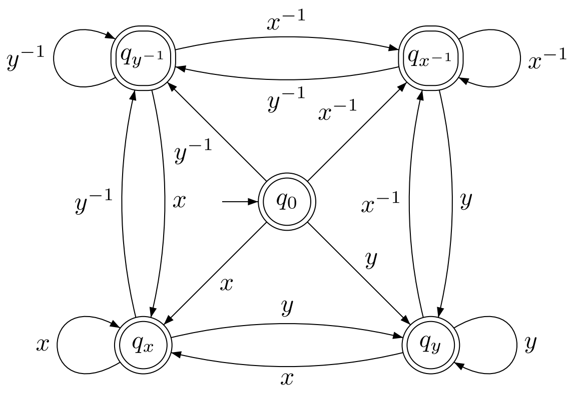

Fig. 1. The finite automaton accepting the reduced words of $F_{\{x,y\}}$.

word $w = \sigma_1\sigma_2\cdots\sigma_n \in \Sigma^*$ is *accepted* by $\mathcal{A}$ if there exists a run ρ of $\mathcal{A}$ on w such that $\rho(n+1) \in F$. In short, $\mathcal{A}$ accepts w if there is a sequence of choices allowed by the transition function such that $\mathcal{A}$ is in a final state after reading the word w. The set of all words $w \in \Sigma^*$ accepted by $\mathcal{A}$ is called the *language accepted* by $\mathcal{A}$ and it is denoted by $L(\mathcal{A})$.

The automaton $\mathcal{A}$ is said to be *deterministic* if $|\delta(q, a)| \leq 1$ for all $q \in Q$ and $a \in \Sigma$, where $|\cdot|$ denotes cardinality.

The following is a fundamental characterization of regular languages (see, e.g. [22,46,49]).

Theorem 2.4. *Let Σ be a finite alphabet and $L \subset \Sigma^*$ be a language. Then L is regular (that is, it is generated by a left-linear (equivalently, by a right-linear) grammar) if and only if it is accepted by a deterministic finite automaton.*

Example 2.5 (*The Free Group*)**.** Let X be a finite set with $\Sigma = X \cup X^{-1}$ and the map $x \mapsto x^{-1}$ as in Example 2.3. Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ be the finite state automaton with state set $Q = \{q_0\} \cup \{q_x : x \in \Sigma\}$, $F = Q$ (all states are terminal), and where the transition function is defined by

$$\delta(q_0, x) = q_x$$
$$\delta(q_x, y) = \begin{cases} q_y & \text{if } y \neq x^{-1} \\ \varnothing & \text{otherwise} \end{cases}$$

for all $x, y \in \Sigma$. It is immediate to see that the language accepted by the automaton $\mathcal{A}$ consists of all reduced words over the alphabet Σ, that is, $L(\mathcal{A}) = L_{\text{red}}(\Sigma)$.

Graphically, one represents a finite automaton $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ as a labelled graph. (See Section 3.1 for more on labelled graphs.) The vertex set is Q and, for every $p \in Q$ and $a \in \Sigma$, there is an oriented edge from p to q, with label a, for all $q \in \delta(p, a)$. The initial state is denoted by an ingoing arrow into it and a double circle is drawn around each final state. In Fig. 1 we represented the automaton $\mathcal{A}$ recognizing the language $L_{\text{red}}(\Sigma)$ of reduced words on $\{x, y, x^{-1}, y^{-1}\}$.

2.5. Pushdown automata

A *pushdown automaton* is a 7-tuple $\mathcal{M} = (Q, \Sigma, Z, \delta, q_0, F, z_0)$, where Q is a nonempty finite set of *states*, Σ is a finite alphabet, called the *input alphabet*, Z is a finite set of *stack symbols*, $q_0 \in Q$ is the *initial state*, $F \subset Q$ is the set of *final states*, and $z_0 \in Z \cup \{\varepsilon\}$ is the *start symbol*. Finally, the *transition function* is a map

$$\delta: Q \times (\Sigma \cup \{\varepsilon\}) \times (Z \cup \{\varepsilon\}) \to \mathcal{P}_{\text{fin}}(Q \times Z^*)$$

where $\mathcal{P}_{\text{fin}}(Q \times Z^*)$ stands for the set of all finite subsets of $Q \times Z^*$.

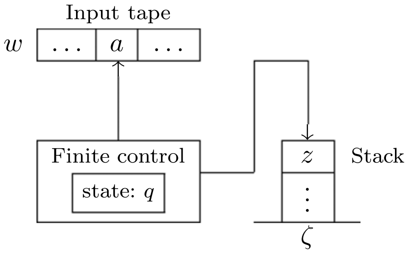

Fig. 2. Representation of a pushdown automaton. The input tape contains the word w and its current letter is a. The stack contains the word ζ starting by the letter z.

The automaton is represented in Fig. 2 and works in the following way. The automaton reads a word $w \in \Sigma^*$ from the input tape, letter by letter, from left to right. At any time, it is in some state $q \in Q$, and the stack contains a word $\zeta \in Z^*$. If the current letter of w is a, the state is q and the top symbol of the stack word ζ is z, then it performs one of the following transitions:

(i) $\mathcal{M}$ moves to the next position on the input tape. If the letter read is a, $\mathcal{M}$ selects some $(q', \zeta') \in \delta(q, a, z)$, changes to state q', and replaces the rightmost symbol z of ζ by ζ'. If there are no more letters on the input tape the machine halts. Or, without advancing the tape,

(ii) $\mathcal{M}$ selects some $(q', \zeta') \in \delta(q, \varepsilon, z)$, changes to state q', remains at the current position on the input tape and replaces the rightmost symbol z of ζ by ζ'. Note that $\mathcal{M}$ can make several successive moves of this type without advancing the tape. Transitions of this type are called *ε-transitions*.

If both $\delta(q, a, z)$ and $\delta(q, \varepsilon, z)$ are empty then $\mathcal{M}$ halts.

Note that, in general, a pushdown automaton is *nondeterministic* in the sense that it has more than one choice of a possible transition. A pushdown automaton $\mathcal{M}$ is *deterministic* if for any $q \in Q$, $a \in \Sigma$ and $z \in Z \cup \{\varepsilon\}$, it has at most one option of what to do next, that is,

$$|\delta(q, a, z)| + |\delta(q, \varepsilon, z)| \leq 1.$$

Since we are interested in groups, our convention is that the automaton is allowed to continue to work when the stack is empty, i.e., when $\zeta = \varepsilon$. Then the automaton acts in the same way as before, by changing to state q' and putting ζ' in the stack if it advances the tape and selects $(q', \zeta') \in \delta(q, a, \varepsilon)$ in case (i), or by making an ε-transition $(q', \zeta') \in \delta(q, \varepsilon, \varepsilon)$ in case (ii). This convention is different from that of many authors, for example [49], who require the automaton to halt on an empty stack.

Let $w \in \Sigma^*$, $q \in Q$, and $\zeta \in Z^*$. We write $\mathcal{M} \overset{*}{\underset{w}{\vdash}} (q, \zeta)$ if, starting at the initial state q_0 and with only z_0 in the stack, it is possible for the automaton $\mathcal{M}$ (after finitely many transitions) to be in state q with ζ written on the stack, after reading the input w. If $q \in F$ and $\zeta = \varepsilon$ we say that $\mathcal{M}$ *accepts* w. The *language accepted* by $\mathcal{M}$ is then defined by

$$L(\mathcal{M}) := \{w \in \Sigma^* : \mathcal{M} \overset{*}{\underset{w}{\vdash}} (q, \varepsilon) \text{ for some } q \in F\}.$$

Example 2.6. Every finite automaton $\mathcal{A}$ may be viewed as a pushdown automaton. Indeed, if $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$, consider the pushdown automaton $\mathcal{M} = (Q, \Sigma, Z, \delta', q_0, F, \varepsilon)$, where $Z = \varnothing$ and the transition function $\delta' : Q \times (\Sigma \cup \{\varepsilon\}) \times \{\varepsilon\} \to \mathcal{P}_{\text{fin}}(Q \times \{\varepsilon\})$ is defined by setting

$$\delta'(q, a, \varepsilon) = \{(q', \varepsilon) : q' \in \delta(q, a)\}$$

for all $q \in Q$ and $a \in \Sigma$. It is clear that $L(\mathcal{A}) = L(\mathcal{M})$. Note that $\mathcal{M}$ is deterministic whenever $\mathcal{A}$ is deterministic.

The following is a fundamental characterization of context-free languages (see [22,46,49]).

Theorem 2.7 (*Chomsky*)**.** *Let Σ be a finite alphabet and $L \subset \Sigma^*$ be a language. Then L is context-free (that is, it is generated by a context-free grammar) if and only if it is accepted by a pushdown automaton. Moreover, L is unambiguous if and only if it is accepted by a deterministic pushdown automaton.*

Note that since there exist inherently ambiguous context-free languages (which therefore are not accepted by any deterministic pushdown automaton), it follows that nondeterministic pushdown automata are strictly more powerful than deterministic ones.

Example 2.8 (*The Dyck Language Revisited*)**.** Let $n \geq 1$ and $\Sigma = \{a_1, \bar{a}_1, a_2, \bar{a}_2, \ldots, a_n, \bar{a}_n\}$. Consider the deterministic pushdown automaton $\mathcal{M} = (Q, \Sigma, Z, \delta, q_0, F, z_0)$ with $Q = \{q_0\} = F, Z = \Sigma$, $z_0 = \varepsilon$ and δ: $\{q_0\} \times (\Sigma \cup \{\varepsilon\}) \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}_{\text{fin}}(\{q_0\} \times \Sigma^*)$ defined by setting

$$\delta(q_0, a, z) = \begin{cases} \{(q_0, \varepsilon)\} & \text{if } a = \bar{z} \\ \{(q_0, za)\} & \text{otherwise} \end{cases}$$

for all $a, z \in \Sigma \cup \{\varepsilon\}$. (We use the convention that $\bar{\varepsilon} = \varepsilon$.) Then it is easy to check that $L(\mathcal{M})$ is the Dyck language defined in Example 2.1.

2.6. Turing machines, computable and computably enumerable languages

One of the great accomplishments of twentieth century mathematics was the formalization of the idea of being "computable". Probably the clearest model is Turing's concept of a *Turing machine* [93], which one can consider as an idealized digital computer. Several other definitions were proposed in the 1930's and 1940's and all of these definitions have been shown to be equivalent. The Turing machine model of computation is the one still used in studying computational complexity, where one wants to investigate how difficult it is to calculate something.

Thesis 2.9 (*The Church–Turing Thesis*)**.** *Any function intuitively thought to be computable is computable by a Turing machine.*

Seventy years of research have led to the general acceptance of the Church–Turing Thesis. By the word "algorithm" we therefore mean a Turing machine.

We give a brief description of how a Turing machine works. This description is illustrated in Fig. 3. For a careful detailed discussion see [22,24,49]. A Turing machine $\mathcal{T}$ consists of the following:

- A *tape* which is divided into consecutive *cells* or *squares* and which is infinite to the right. Thus the Turing machine always has enough tape for any computation, that is, it has unlimited memory. There is a *tape alphabet* Γ which contains a special *blank symbol* b. The *input alphabet* is $\Sigma \subset \Gamma \setminus \{b\}$. Each cell contains a symbol from the tape alphabet and initially, all but finitely many cells contain the blank symbol b.
- A *reading head* that can read and write symbols on the tape and then move one cell to the right or one cell to the left. Symbols L and R stand for "left" and "right", respectively.
- A finite set Q of *control states* with an *initial state* $q_0 \in Q$ and a *halting state* $H \in Q$.
- A *program* or *transition function* $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$. There is only one type of instruction and Turing machines are thus the ultimate in "reduced instruction set architecture". If $\delta(q, \gamma) = (q', \gamma', L/R)$ then the machine immediately halts if $q' = H$. Otherwise the machine does the following operations in sequence:
  - replace the symbol γ by the symbol γ', which may be the same as γ or may be the blank b,
  - move the reading head one cell to the left (on L) or one cell to the right (on R),
  - assume the new state $q' \in Q$.

A word $w \in \Sigma^*$ is *written* on the tape if it occupies the leftmost cells of the tape. It is understood that all the cells that are on the right of the cell containing the last letter of w contain the blank symbol b.

Turing machines can be regarded either as *calculators* of functions or as *enumerators*.

Definition 2.10. Let Σ_1 and Σ_2 be finite alphabets. A function $f : \Sigma_1^* \to \Sigma_2^*$ is *computable* if there exists a Turing machine $\mathcal{T}$ which, when started in its initial state with the reading head at the left end of the tape and a word $w \in \Sigma_1^*$ written on the tape, eventually halts with $f(w) \in \Sigma_2^*$ written on the tape.

A set $L \subseteq \Sigma^*$ is *computable* if its characteristic function $\chi_L : \Sigma^* \to \{0, 1\}$ is computable.

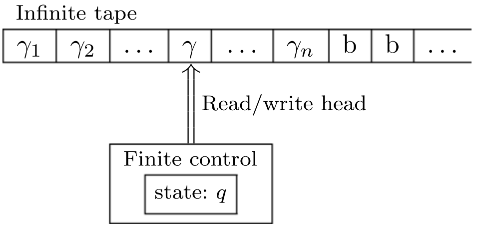

Fig. 3. Representation of a Turing machine.

Note that a Turing machine which calculates a function is required to halt on all inputs. In general, a Turing machine with input alphabet Σ may not halt on all inputs.

Definition 2.11. A set $L \subseteq \Sigma^*$ is *computably enumerable* if there exists a Turing machine $\mathcal{T}$ with input alphabet Σ such that $\mathcal{T}$ halts on input w if and only if $w \in L$. We say that $\mathcal{T}$ *enumerates* or *accepts* L.

Thus computably enumerable languages are exactly the halting sets of Turing machines. The following lemma is a basic fact about computability.

Lemma 2.12. *A set $L \subseteq \Sigma^*$ is computable if and only if both L and its complement $\neg L = \Sigma^* \setminus L$ are computably enumerable.*

Proof. A basic principle of constructing Turing machines is that a Turing machine $\mathcal{T}$ can be always used as a subroutine in a larger machine $\widehat{\mathcal{T}}$. If L is computable, let $\mathcal{T}$ compute the characteristic function χ_L of L. The machine $\widehat{\mathcal{T}}$ enumerating L works as follows. On input w, the machine $\widehat{\mathcal{T}}$ uses $\mathcal{T}$ to compute $\chi_L(w)$. If $w \in L$ then $\widehat{\mathcal{T}}$ halts. If $w \notin L$ then $\widehat{\mathcal{T}}$ goes into a loop and never halts. The machine enumerating the complement $\neg L$ works similarly.

Conversely, suppose that $\mathcal{T}_1$ and $\mathcal{T}_2$ enumerate L and $\neg L$ respectively. The machine $\overline{\mathcal{T}}$ computing L uses the basic technique of "bounded simulation". On input w, the machine $\overline{\mathcal{T}}$ begins successively enumerating positive integers n. When n is enumerated, $\overline{\mathcal{T}}$ simulates both $\mathcal{T}_1$ and $\mathcal{T}_2$ on input w for n steps and sees if either machine halts in n steps. Since L and $\neg L$ are complements, exactly one of $\mathcal{T}_1$ or $\mathcal{T}_2$ will eventually halt on input w. When one of them halts, $\overline{\mathcal{T}}$ then erases its tape and writes 1 if $\mathcal{T}_1$ halted and 0 if $\mathcal{T}_2$ halted. $\square$

Note that in order to be able to prove that a problem is *not* computable, it is necessary to have a complete list of all possible means of computation. We can assume that the input alphabet of a Turing machine contains the symbols 0 and 1. It is not difficult to effectively assign a unique binary number $g(\mathcal{T})$ to each Turing machine $\mathcal{T}$ (see [49]). The *Halting Problem* for Turing machines is the following problem: given a Turing machine $\mathcal{T}$ and an input $w \in \{0, 1\}^*$, does the machine $\mathcal{T}$ halt on input w? Turing [93] showed that the Halting Problem is not computable. Once one has a non-computable language L, one can use "reduction" to show that a language L' is not computable by showing that L is reducible to L' in the sense that if L' were computable then L would be computable. All non-computability results eventually go back to the Halting Problem.

3. Finitely generated groups, Cayley graphs, and the Word Problem

3.1. Labelled graphs

A *labelled graph* is a triple $\Gamma = (V, E, \Sigma)$, where $V = V(\Gamma)$ is the set of *vertices*, Σ is a finite alphabet, and $E = E(\Gamma) \subset V \times \Sigma \times V$ is the set of *oriented, labelled edges*.

Let $\Gamma = (V, E, \Sigma)$ be a labelled graph.

We say that Γ is *finite* if its vertex set V is finite and thus the edge set E is also finite.

Given an edge $e = (u, a, v) \in E$ its *label* is $\lambda(e) := a \in \Sigma$, its *initial vertex* is $o(e) := u \in V$, and its *terminal vertex* is $t(e) := v \in V$. We say that e is *outgoing* from u and *ingoing* into v. An edge e can be visualized as an arrow from $o(e)$ to $t(e)$.

For $v \in V$ we denote by $\partial^o(v) \in [0, \infty]$ (resp. $\partial^t(v) \in [0, \infty]$) the number (possibly infinite) of edges outgoing from (resp. ingoing into) v. The quantity $\partial(v) = \partial^o(v) + \partial^t(v) \in [0, \infty]$ is the *degree* of v. An edge of the form (v, a, v) is called a *loop* at v and is both an outgoing edge and an ingoing edge at v, and so contributes 2 to $\partial(v)$. If $\partial(v) < \infty$ for all $v \in V$ one says that Γ is *locally finite*. If the degrees of the vertices of Γ are uniformly bounded, that is $\sup_{v\in V} \partial(v) < \infty$, one says that Γ has *bounded degree*.

Suppose that Σ is equipped with an involution $a \mapsto \bar{a}$. We then say that Γ is *symmetric* if for each edge $e = (u, a, v) \in E$, the *inverse edge* $e^{-1} = (v, \bar{a}, u)$ also belongs to E. The drawing convention for symmetric graphs is that one draws only one directed edge (with the corresponding label) choosing between e and e^{-1}.

Note that if Γ is symmetric, we clearly have $\partial^o(v) = \partial^t(v)$ for each $v \in V$. If, in addition, there exists $d \in \mathbb{N}$ such that $d = \partial^o(v) = \partial^t(v)$ for all $v \in V$, one says that Γ is *regular* of *degree d*.

We say that Γ is *deterministic* if at every vertex all outgoing edges have distinct labels.

Note that our definition of a labelled graph allows *multiple edges*, i.e., distinct edges of the form $e_1 = (u, a_1, v)$ and $e_2 = (u, a_2, v)$, but this implies that $a_1 \neq a_2$. Thus, two edges must coincide if they have the same initial vertex, the same terminal vertex, and the same label.

A *subgraph* of Γ is a labelled graph $\overline{\Gamma} = (\overline{V}, \overline{E}, \overline{\Sigma})$ such that $\overline{V} \subset V$, $\overline{E} \subset E$ and $\overline{\Sigma} \subset \Sigma$.

Let $\Gamma' = (V', E', \Sigma)$ be another labelled graph with the same label alphabet Σ. A *labelled graph-homomorphism* from Γ to Γ' is a map $\varphi: V \to V'$ such that $(\varphi(u), a, \varphi(v)) \in E'$ for all $(u, a, v) \in E$. A *labelled graph-isomorphism* from Γ to Γ' is a bijective labelled graph-homomorphism from Γ to Γ' such that the inverse map $\varphi^{-1}: V' \to V$ is also a labelled graph-homomorphism from Γ' to Γ. Note that if φ is a labelled graph-isomorphism from Γ to Γ', then the map $\psi: E \to E'$ defined by $\psi(u, a, v) = (\varphi(u), a, \varphi(v))$, for all $(u, a, v) \in E$, is bijective with inverse map $\psi^{-1}: E' \to E$ given by $\psi^{-1}(u', a, v') = (\varphi^{-1}(u'), a, \varphi^{-1}(v'))$, for all $(u', a, v') \in E'$.

A *path* in Γ is a sequence $\pi = (e_1, e_2, \ldots, e_n)$ of edges such that $o(e_{i+1}) = t(e_i)$ for $i = 1, 2, \ldots, n-1$. We extend our notation for initial and terminal vertices to paths. The vertex $o(\pi) := o(e_1)$ is the *initial* vertex of π and $t(\pi) := t(e_n)$ is the *terminal* vertex of π. We then say that π *starts* at $o(\pi)$ and *ends* at $t(\pi)$, equivalently it *connects* $o(\pi)$ to $t(\pi)$. An edge $e \in E$ such that $o(e) = t(e)$ is called a *loop*. For every vertex $v \in V$, we also allow the *empty path* starting and ending at v.

One says that Γ is *strongly connected* provided that for all vertices $u, v \in V$ there exists a path connecting u to v. If Γ is symmetric, the (obviously reflexive and transitive) relation in V defined by $u \sim v$ provided that there exists a path in Γ connecting u to v is also symmetric and therefore an equivalence relation. Then the corresponding equivalence classes are called the *connected components* of Γ; clearly, Γ is strongly connected if and only if there exists a unique such a connected component.

Let $\pi = (e_1, e_2, \ldots, e_n)$ be a path. The number $|\pi| = n$ of edges is the *length* of the path. The *label* of π is $\lambda(\pi) := \lambda(e_1)\lambda(e_2)\cdots\lambda(e_n) \in \Sigma^*$. The empty path has length 0 and is labelled by the empty word ε. If $t(\pi) = o(\pi)$ one says that π is *closed*. If the vertices $o(e_1), t(e_1), t(e_2), \ldots, t(e_n)$ are all distinct, then the path is called *simple*. If π is closed, contains an edge and its vertices are all distinct with the exception of $o(e_1) = t(e_n)$, then π is called a *cycle*.

Denote by $\Pi_{u,v}(\Gamma)$ the set of all paths π in Γ with initial vertex $o(\pi) = u$ and terminal vertex $t(\pi) = v$. More generally, given a subset $F \subset V$ we set $\Pi_{u,F}(\Gamma) := \bigcup_{v\in F} \Pi_{u,v}(\Gamma)$. For $u \in V$ and $F \subset V$ we define the language

$$L_{u,F}(\Gamma) := \{\lambda(\pi) : \pi \in \Pi_{u,F}(\Gamma)\} \subset \Sigma^*.$$

Note that $L_{u,F}(\Gamma)$ may be empty.

Suppose that a given vertex $v_0 \in V$ of Γ is fixed as *origin* (or *root* or *basepoint*). One then says that $\Gamma = (V, E, \Sigma, v_0)$ is a *rooted* labelled graph. A *rooted* labelled graph-homomorphism (resp. *rooted* labelled graph-isomorphism) from a rooted labelled graph Γ into a rooted labelled graph Γ' is a labelled graph-homomorphism (resp. labelled graph-isomorphism) $\varphi: V \to V'$ such that $\varphi(v_0) = v_0'$, where $v_0' \in V'$ is the root of Γ'.

Example 3.1 (*The Rooted Infinite Binary Tree T_2*)**.** Let $\Sigma = \{0, 1\}$. Consider the rooted labelled graph $\Gamma = (\Sigma^*, E, \Sigma, \varepsilon)$ where $E = \{(v, a, va) : v \in \Sigma^*, a \in \Sigma\}$. The vertex corresponding to the empty word ε is the root of Γ. Note that for every vertex $v \in V$ one has $\partial^o(v) = 2$.

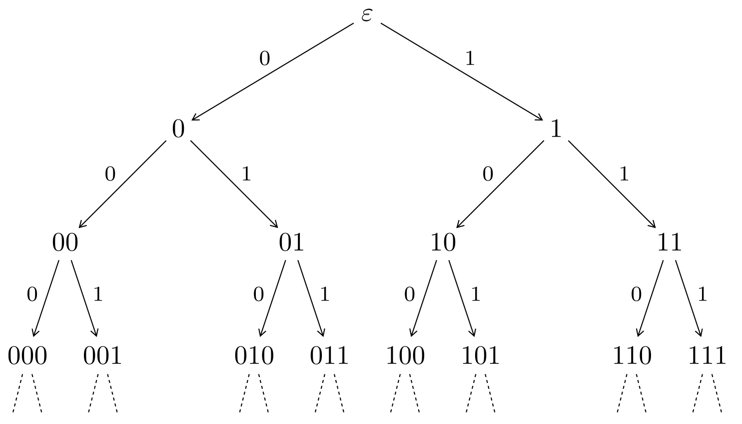

Fig. 4. The rooted infinite binary tree T_2.

The graph Γ is a rooted, directed tree called the *rooted infinite binary tree* and it is denoted by T_2. Fig. 4 illustrates it.

Example 3.2 (*The Graph Underlying a Finite State Automaton*)**.** Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ be a finite state automaton. Consider the labelled graph $\Gamma = (V, E, \Sigma)$ where $V = Q$ and $E \subset V \times \Sigma \times V$ is defined by

$$E = \{(u, a, v) : u \in V, a \in \Sigma, \text{ and } v \in \delta(u, a)\}.$$

Note that Γ is deterministic if and only if $\mathcal{A}$ is deterministic. The language $L(\mathcal{A}) \subset \Sigma^*$ accepted by $\mathcal{A}$ can be reinterpreted as the language consisting of all words of the form $\lambda(\pi)$, where π is a path in Γ starting at the initial state q_0 and terminating at some final state in F. In symbols:

$$L(\mathcal{A}) = L_{q_0,F}(\Gamma) = \{\lambda(\pi) : \pi \in \Pi_{q_0,F}(\Gamma)\}.$$

3.2. Presentations and Cayley graphs

A *finitely generated group presentation* is a pair $\langle X; R\rangle$, where X is a finite set of *generators*, the *group alphabet* is $\Sigma = X \cup X^{-1}$ where X^{-1} is a disjoint copy of X, and the set R of *defining relators* is a subset of Σ^* (cf. [65,62]). We denote by $a \mapsto a^{-1}$ the involutive map on Σ exchanging X and X^{-1}.

Two words $u, v \in \Sigma^*$ are said to be *equivalent*, written $u \approx v$, if it is possible to transform u into v by a finite sequence of insertions or deletions of either the defining relators $r \in R$ or the *trivial relators* of the form xx^{-1} and $x^{-1}x$, with $x \in X$. The concatenation product on the free monoid Σ^* (cf. Equation (2.1)) induces a group structure on the set $G = \Sigma^*/\approx$ of equivalence classes whose identity element is the class of the empty word ε. Moreover, if $w = a_1a_2 \cdots a_n \in \Sigma^*$, the inverse of the class of w is the class of the element $w^{-1} \in \Sigma^*$ defined by $w^{-1} = a_n^{-1} \cdots a_2^{-1}a_1^{-1}$. One says that $\langle X; R\rangle$ is a *presentation* of the group G and one writes $G = \langle X; R\rangle$. When the defining relators $r \in R$ are of the form $r = u_r v_r^{-1}$ for some $u_r, v_r \in \Sigma^*$ one often writes $G = \langle X; u_r = v_r, r \in R\rangle$ and refers to the equations $u_r = v_r, r \in R$, as the *defining relations*.

A presentation $\langle X; R\rangle$ where both X and the set R of relators is finite is called a *finite presentation*. A group admitting a finite presentation is called *finitely presentable*.

Given a presentation $G = \langle X; R\rangle$, if F_X denotes the free group based on X and N is the normal closure of R in F_X then the group homomorphism $F_X \to G$ sending each $x \in X$ to its $\approx$-equivalence class in Σ^* induces a group isomorphism $F_X/N \to G$.

Example 3.3. (a) Let $G = \{g_1, g_2, \ldots, g_n\}$ be a finite group where g_1 is the identity element. The *multiplication table presentation* of G is the presentation

$$G = \langle g_2, g_3, \ldots, g_n; g_ig_j = g_{k(i,j)},\ i, j = 2, 3, \ldots, n\rangle,$$

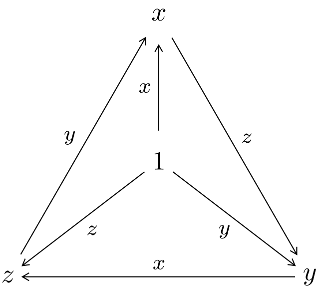

Fig. 5. The Cayley graph of the Klein 4-group $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ with respect to the multiplication table presentation $\langle x, y, z;\ x^2 = y^2 = z^2 = 1,\ xy = z = yx,\ xz = y = zx,\ yz = x = zy\rangle$.

where $g_{k(i,j)}$ is the product of g_i and g_j determined from the multiplication table of G. This example shows that every finite group has a finite presentation.

(b) In multiplicative notation, the infinite cyclic group has a presentation $\mathbb{Z} = \langle x\rangle$ with one generator and no defining relations.

(b′) More generally, the free group based on a finite set X has a presentation $F_X = \langle X\rangle$ with generating set X and no defining relations.

(c) In multiplicative notation, the free abelian group of rank two has a presentation

$$\mathbb{Z}^2 = \langle x, y;\ [x, y]\rangle,$$

where $[x, y] = x^{-1}y^{-1}xy$ is the commutator of x and y.

(c′) More generally, the free abelian group based on a finite set X has presentation

$$\langle X;\ [x, y],\ x, y \in X\rangle.$$

In this case, the normal closure N of $R = \{[x, y], x, y \in X\}$ in the free group F_X based on X is the *commutator* (or *derived*) subgroup of F_X.

(d) Let $G = (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z})$ be the free product of four copies of the group $\mathbb{Z}/2\mathbb{Z}$ with two elements. Let x, y, z and w be the nontrivial elements in each copy of $\mathbb{Z}/2\mathbb{Z}$ (so that $x = x^{-1}, y = y^{-1}$, etc). Then the corresponding presentation is $G = \langle x, y, z, w;\ x^2, y^2, z^2, w^2\rangle$.

The fundamental geometric object associated with a finitely generated group was the graph defined by Cayley [13] in 1878 (see also [89]).

Definition 3.4 (*Cayley Graph*)**.** Let $G = \langle X; R\rangle$ be a finitely generated group. The *Cayley graph* of G with respect to the presentation $\langle X; R\rangle$ is the labelled graph $\Gamma = \Gamma(G : X; R)$ whose vertex set is $V(\Gamma) = G$, the set of labelled, directed edges is

$$E(\Gamma) = \{(g, x, gx) :\ g \in G,\ x \in \Sigma\},$$

and the label alphabet is $\Sigma = X \cup X^{-1}$.

Let $\Gamma = \Gamma(G : X; R)$ be a Cayley graph. Then Γ is often regarded as a rooted graph with basepoint $v_0 = 1_G$ and is strongly connected: between any two vertices u and v there is at least one path from u to v. Note that a word $w \in \Sigma^*$ labels a closed path in $\Gamma(G : X; R)$ if and only if w represents the identity in G. Moreover, Γ is symmetric (with respect to the involution $a \mapsto a^{-1}$ on Σ) and $|X|$-regular. If $h \in G$ then the map $\mu_h : G \to G$, defined by $\mu_h(g) = hg$ for all $g \in G$ is a labelled graph automorphism of Γ. Thus a Cayley graph is *homogeneous* in the sense that given any two vertices there is a labelled graph automorphism taking the first vertex to the second.

Example 3.5. (a) In Fig. 5 we illustrate the Cayley graph of the Klein 4-group $\mathbb{Z}/2\mathbb{Z} \times \mathbb{Z}/2\mathbb{Z}$ with respect to the multiplication table presentation $\langle x, y, z;\ x^2 = y^2 = z^2 = 1,\ xy = z = yx,\ xz = y = zx,\ yz = x = zy\rangle$.

(b) The Cayley graph $\Gamma(\mathbb{Z} : x)$ is described in Fig. 6.

(b′) The Cayley graph $\Gamma(F_2 : x, y)$ is described in Fig. 7.

$$\cdots\cdots\; x^{-2} \xrightarrow{x} x^{-1} \xrightarrow{x} 1 \xrightarrow{x} x \xrightarrow{x} x^{2} \;\cdots\cdots$$

Fig. 6. The Cayley graph of the group $\mathbb{Z} = \langle x \rangle$.

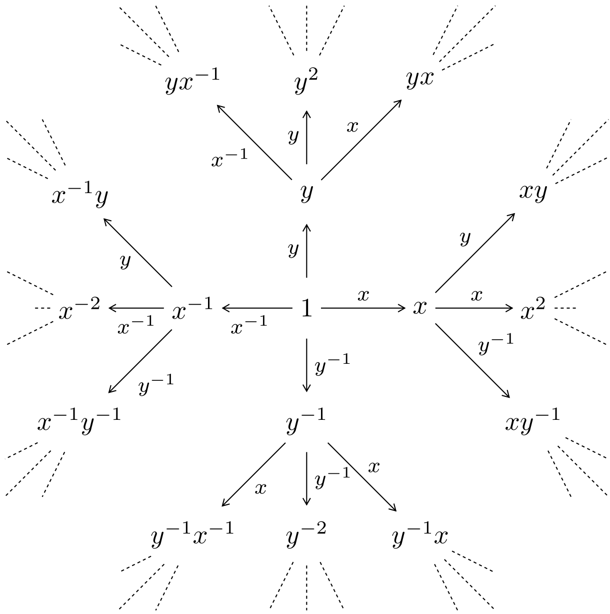

Fig. 7. The Cayley graph of the free group $F_2 = \langle x, y \rangle$.

(c) The Cayley graph $\Gamma(\mathbb{Z}^2 : x, y; [x, y])$ is described in Fig. 8.
(d) The Cayley graph $\Gamma(G : x, y, z, w; x^2, y^2, z^2, w^2)$, where $G = (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z})$, is described in Fig. 9.

Note that the Cayley graphs in (b) and (d) are 4-regular trees and they are isomorphic as directed graphs. However they are not isomorphic as directed labelled graphs.

Let $G = \langle X; R \rangle$ be a finitely generated presentation and let $\Gamma = \Gamma(G : X; R)$ be the corresponding Cayley graph. When equipped with the metric dist: $V \times V \to [0, \infty)$ defined by $\mathrm{dist}(u, v) = \min\{|\pi| : \pi \in \Pi_{u,v}\}$, Γ is a discrete metric space. Denote by $B_n = \{g \in G : \mathrm{dist}(g, 1_G) \leq n\}$ the *ball* of radius n centred at 1_G. The map $\gamma = \gamma(G : X; R) \colon \mathbb{N} \to \mathbb{N}$ defined by $\gamma(n) = |B_n|$ for all $n \in \mathbb{N}$ is called the *growth function* of G with respect to the given presentation. Since γ is subadditive (i.e. $\gamma(n + m) \leq \gamma(n)\gamma(m)$ for all $n, m \in \mathbb{N}$), by a well known result of Fekete the limit

$$\lambda = \lambda(G : X; R) = \lim_{n\to\infty} \sqrt{\gamma(n)},$$

exists and $1 \leq \lambda < \infty$. This limit is called the *growth rate* of G with respect to the given presentation,

That $\lambda = 1$ is a condition independent of the particular presentation. If $G = \langle X'; R' \rangle$ is another finitely generated presentation of G and γ' is the corresponding growth function, then $\lambda' = \lim_{n\to\infty} \sqrt{\gamma'(n)}$ equals 1 if and only if λ does. If $\lambda = 1$ one says that the group G has *subexponential growth*. Otherwise, the group G is said to have *exponential growth*. All finite groups, all finitely generated abelian groups, and, more generally, all nilpotent groups have subexponential growth. On the other hand, if $F_X = \langle X \rangle$ is a finitely generated free group, then $\lambda = 2|X| - 1$ so that F_X has exponential growth if $|X| \geq 2$.

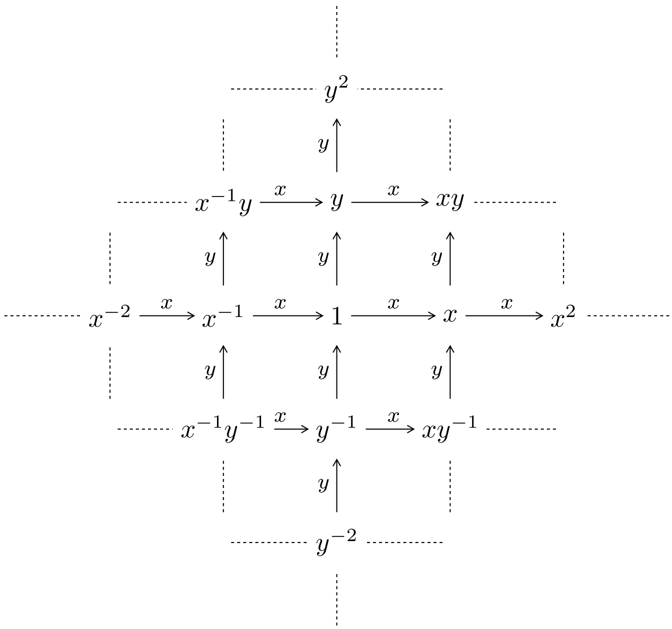

Fig. 8. The Cayley graph of the group $\mathbb{Z}^2 = \langle x, y;\ xy = yx \rangle$.

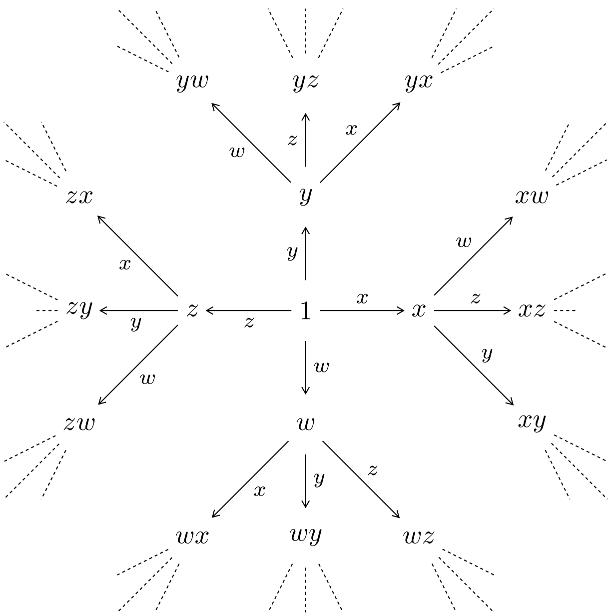

Fig. 9. The Cayley graph of the group $(\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/2\mathbb{Z}) = \langle x, y, z, w;\ x^2 = y^2 = z^2 = w^2 = 1 \rangle$.

3.3. The Word Problem

In a remarkable paper in 1911, twenty years before the development of the theory of computability, Dehn [26] posed three fundamental decision problems in group theory: the *Word Problem*, the *Conjugacy Problem*, and the *Isomorphism Problem*. (See also the expository article by de la Harpe [29].) Dehn viewed the Word Problem as the following algorithmic problem: given a finitely generated group presentation $G = \langle X; R\rangle$ find an algorithm which, when given a word $w \in \Sigma^*$, decides, in a finite number of steps, whether or not w represents the identity element of G. In 1912 Dehn [27] solved this problem for the *fundamental group of a closed orientable surface*:

$$G_h = \left\langle a_1, b_1, a_2, b_2, \ldots, a_h, b_h; \prod_{i=1}^{h}[a_i, b_i]\right\rangle$$

where $h \geq 2$ is the genus of the surface.

Given a finite group presentation $G = \langle X; r_1, \ldots, r_k\rangle$, let R be the *symmetrized* set generated by the given relators, that is, R consists of all cyclic permutations of the r_i and their inverses. Then $\langle X; R\rangle$ is also a presentation of G. The original presentation is a *Dehn presentation* if every nontrivial word w equal to the identity in G contains a subword u such that some $r \in R$ has the form $r = uv$ where $|u| > |v|$. This says that every nontrivial word equal to the identity contains more than half of a cyclic permutation of the given relators or their inverses.

Although we usually do not write the trivial relators, if X is a finite set and F_X is the free group based on X, then a Dehn presentation of F_X is given by $\langle X; xx^{-1}, x^{-1}x, x \in X\rangle$.

Now, every group admitting a Dehn presentation has solvable Word Problem. Indeed, if $G = \langle X; r_1, \ldots, r_k\rangle$ is a Dehn presentation, let R be the symmetrized set of relators generated by the r_i. We then have the following algorithm, now called *Dehn's algorithm*, to decide whether or not $w \in \Sigma^*$ represents the identity element in G:

Step 1. if $w = \varepsilon$ then w does represent 1_G, otherwise go to the next step;

Step 2. if w contains a subword u where for some $r \in R$, $r = uv$ with $|u| > |v|$, then replace u by v^{-1} and go to Step 1. Otherwise, w does not represent 1_G.

Note that since each step in the algorithm strictly reduces the length of the word being considered, Dehn's algorithm takes only linearly many steps and thus works in linear time, which is the best possible complexity result. The Cayley graph of the surface group G_h, $h \geq 2$ is the dual graph of the regular tessellation of the hyperbolic plane by $4h$-gons. Dehn used hyperbolic geometry to show that the presentation of G_h given above is a Dehn presentation and thus G has solvable Word Problem. The quest to extend Dehn's algorithm to a larger class of groups led to the development of *small cancellation theory* which, among many other things, gives some simple sufficient conditions for a presentation to be a Dehn presentation. (See [88] for a survey.) This then led to Gromov's [42] remarkable development of the theory of *word-hyperbolic* groups. As mentioned before, the Cayley graph of a finitely generated group becomes a metric space by defining the distance between two vertices as the minimal length of a path connecting them and considering each edge as isometric to the unit interval. The *thin triangle condition* then captures many of the features of hyperbolic geometry. One of the characterizations of a group G being word-hyperbolic is exactly that it has some Dehn presentation. (See [42] and also [10, Chapter III.Γ, Theorem 2.6.].)

Solvability of the Word Problem was extended to all *one-relator groups* by Magnus [64] in 1932. We do not, however, know any bound on the complexity of solving the Word Problem over the class of all one-relator groups. A theorem of Newman [83,62] shows that any one-relator presentation of the form $G = \langle X; w^n\rangle$ with $n \geq 2$ is a Dehn presentation.

It was independently shown by Novikov [84] in 1955 and by Boone [8] in 1958 that there exist finitely presented groups $G = \langle X; R\rangle$ with unsolvable Word Problem. In order to prove this basic result it is necessary to code the Halting Problem for Turing machines into the Word Problem of the group. The unsolvability of the Word Problem is the foundation of all the unsolvability results in group theory and topology.

3.4. The Dehn function

Let

$$G = \langle X; R \rangle \tag{3.1}$$

be a finite presentation of G. Let $\Sigma = X \cup X^{-1}$ denote the associated group alphabet and suppose that $w \in \Sigma^*$ satisfies $w \approx \varepsilon$, that is, $w = 1_G$ in G. This is equivalent to saying that the reduced form of w belongs to the normal closure N of R in F_X, the free group based on X. This in turn is equivalent to the existence of an expression

$$w = u_1 r_1 u_1^{-1} u_2 r_2 u_2^{-1} \cdots u_m r_m u_m^{-1} \tag{3.2}$$

where $m \in \mathbb{N}$, $u_i \in \Sigma^*$ and $r_i \in R^{\pm 1}$, $i = 1, 2, \ldots, m$. Then the *area* of w (with respect to the given presentation (3.1)), denoted $\mathrm{Area}(w)$, is the smallest $m \geq 0$ such that an expression of the form above holds. The *Dehn function* associated with the presentation (3.1) is the map $\mathrm{Dehn}\colon \mathbb{N} \to \mathbb{N}$ defined by

$$\mathrm{Dehn}(n) = \max\{\mathrm{Area}(w) : w \in \Sigma^*, w \approx \varepsilon, |w| \leq n\}.$$

Cannon observed the following (see also [37, Theorem 2.1]).

Theorem 3.6. *A finitely presented group presentation $G = \langle X; R \rangle$ has a computable Dehn function if and only if the group G has solvable Word Problem.*

It is not difficult to show that if $w = 1_G$ in G then in an expression (3.2) the length of all the conjugating elements u_i can be bounded by $|w|$. Thus if we can calculate $\mathrm{Dehn}(|w|) = b$ we can try all possible products of the form (3.2) with $m \leq b$ and all $|u_i| \leq |w|$ and check whether any of these products equals w in the free group.

3.5. The Word Problem as a formal language

Anisimov [2] in 1972 introduced the fruitful idea of viewing the Word Problem as a formal language, a point of view which we now adopt.

Definition 3.7. Let $G = \langle X; R \rangle$ be a finitely generated group presentation. The *Word Problem* of G, relative to the given presentation, is the language

$$\mathrm{WP}(G : X; R) = \{w \in \Sigma^* : w \approx \varepsilon\},$$

where Σ is the group alphabet as usual. One says that the G has *regular* (resp. *context-free*, resp. *computable*) Word Problem with respect to the given presentation if $\mathrm{WP}(G : X; R)$ is a regular (resp. context-free, resp. computable) language.

Note that, the Word Problem for a finitely generated group presentation $G = \langle X; R \rangle$ is solvable (in the sense of Dehn) if and only if the language $\mathrm{WP}(G : X; R) \subset \Sigma^*$ is computable.

Observation 3.8 (*Invariance and Finitely Generated Subgroups*)**.** It is easy to see that the classification above of the Word Problem as a formal language is actually a property of the group and does not depend on the particular presentation considered. Indeed, the complexity of the Word Problem of a finitely generated group bounds the complexity of the Word Problems of all its finitely generated subgroups. For this, suppose that $G = \langle X; R \rangle$ has a Word Problem of a given type and that $H = \langle Y; S \rangle$ is a finitely generated presentation of a group isomorphic to a finitely generated subgroup of G. Let $\phi : H \to G$ be an injective homomorphism and for $y \in Y$ denote by $w_y \in \Sigma^*$ a representative of the image $\phi(y)$. So, whether a finite automaton, pushdown automaton or Turing machine $\mathfrak{M}$ accepts the Word Problem for the first presentation, we can construct a machine $\mathfrak{M}'$ of the same type which, on reading a letter $(y)^{\pm 1} \in Y \cup Y^{-1}$ simulates the sequence of transitions of $\mathfrak{M}$ on reading the word $(w_y)^{\pm 1} \in \Sigma^*$.

As a consequence, we say that a finitely generated group G is *context-free* provided that the Word Problem $\mathrm{WP}(G : X; R)$ relative to some (equivalently, every) finitely generated presentation $G = \langle X; R \rangle$ is context-free.

Anisimov [2] characterized groups with regular Word Problem.

Theorem 3.9 (*Anisimov*). *Let $G = \langle X; R\rangle$ be a finitely generated group. Then G has regular Word Problem if and only if G is finite.*

Proof. Suppose that G is finite. Consider the deterministic finite automaton $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ where $Q = G$, 1_G is both the initial state q_0 and the unique element in F, and $\delta: Q \times \Sigma \to \mathcal{P}(Q)$ is given by

$$\delta(q, a) = qa$$

for all $q \in Q$ and $a \in A$. Note that the graph underlying $\mathcal{A}$ (cf. Example 3.2) is the Cayley graph $\Gamma(G; X; R)$. Then $\mathcal{A}$ accepts exactly the Word Problem of G.

Conversely, if G is infinite there are arbitrarily long words $w \in \Sigma^*$ such that no nontrivial subword of w is equal to the identity in G. Suppose that $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ is a deterministic finite automaton with alphabet the group alphabet Σ and let n denote the cardinality of its state set Q. Taking a word w as above and such that $|w| \geq n + 1$, then there exist $q \in Q$ and words $w_1, w_2, w_3 \in \Sigma^*$ satisfying $w = w_1w_2w_3$ with $w_2 \neq \varepsilon$ such that $\mathcal{A}$, when reading w, is in the same state q after reading the initial segment w_1 and after reading w_1w_2. Then if $\mathcal{A}$ is in the state $q' \in Q$ after reading $w_1w_1^{-1}$ it is in the same state after reading $w_1w_2w_1^{-1}$. But the first word equals the identity in G while the second word does not. It follows that $\mathcal{A}$ cannot accept the Word Problem of G. $\square$

3.6. *Context-free groups*

Recall that a group is context-free if it is finitely generated and its Word Problem with respect to some (equivalently, every) finitely generated presentation is a context-free language.

Example 3.10 (*The Word Problem for the Free Group*). Let X be a finite set and let $G = F_X$ be the free group based on X. Recall from Example 2.3 that G is in one-to-one correspondence with the set $L_{\text{red}}(\Sigma)$ of all reduced words over the alphabet $\Sigma = X \cup X^{-1}$. We adopt the convention that $\varepsilon^{-1} = \varepsilon$. Consider the one-state deterministic pushdown automaton $\mathcal{M} = (\{q_0\}, \Sigma, \Sigma, \delta, q_0, \{q_0\}, \varepsilon)$. The automaton starts with empty stack, accepts by empty stack, and the transition function is defined by

$$\delta(q_0, a, z) = \begin{cases} (q_0, \varepsilon) & \text{if } a = z^{-1} \\ (q_0, za) & \text{otherwise} \end{cases}$$

for all $a, z \in \Sigma$. It is clear that $L(\mathcal{M}) = \mathrm{WP}(G : X; R)$, so the Word Problem for G is context-free. It follows that free groups are context-free.

For the next example we need the following well-known result (see [49, Lemma 6.1]).

Lemma 3.11 (*The Pumping Lemma for Context-Free Languages*). *Let Σ be a finite alphabet. Let $L \subset \Sigma^*$ be a context-free language. Then there exists a positive integer $N = N(L)$ such that if $w \in L$ and $|w| \geq N$, then we can find $u, v, z, s, t \in \Sigma^*$ such that $w = uvzst$, $|v| + |s| \geq 1$, $|vzs| \leq N$ and $uv^nzx^ny \in L$ for all $n \geq 0$.*

With the notation from the above lemma, we say that the word uv^nzs^nt is obtained from w by *pumping* the subwords v and s.

Example 3.12 (*The Word Problem for the Free Abelian Group of Rank 2*). Let $G = \mathbb{Z}^2$ with presentation $\langle x, y; [x, y]\rangle$. Then $x^my^mx^{-n}y^{-n} = 1$ in G if and only if $m = n$. We can now use Lemma 3.11 to show that $L = \mathrm{WP}(G : X; R)$ is not context-free. Suppose by contradiction that L is context-free and let $N = N(L)$ be the corresponding positive integer. Consider the word $w = x^{N+1}y^{N+1}x^{-(N+1)}y^{-(N+1)}$. We clearly have $w \in L$. However, there are no subwords u, v, z, s, t of w satisfying the conditions described in the Pumping Lemma. Indeed, from $|vzs| \leq N$ we deduce that vzs is a subword of one of the following forms: (i) x^m, (ii) x^py^q, (iii) y^hx^{-k}, (iv) $x^{-p}y^{-q}$, or (v) y^{-m}, for suitable positive integers m, p, q, h and k. In all these cases, by pumping $n \geq 2$ times the subwords v and s, we obtain a word w'

whose number of positive occurrences of x or of y fails to equal the number of its negative occurrences so that $w' \notin L$, contradicting the Pumping Lemma. It follows that L is not context-free. Therefore $\mathbb{Z}^2$ is not a context-free group.

Proposition 3.13. *Let G be a finitely generated group and H a subgroup with $[G : H] < \infty$. Then G is context-free if and only if H is context-free.*

Proof. We have already seen the "only if" part in Observation 3.8. Conversely, let H be a finite index subgroup of G and suppose that it is context-free. Recall the following general fact from group theory (sometimes called the Poincaré Lemma): a subgroup of finite index in a finitely generated group G contains a subgroup which is normal in G and also of finite index, and which is therefore finitely generated. Thus this normal subgroup is also context-free if the ambient subgroup is context-free. We can therefore suppose that H is normal in G. Let $K = G/H$ be the corresponding finite quotient with $\psi: G \to K$ the natural quotient map. Let $K = \{k_1 = 1, k_2, \ldots, k_n\}$. Let $H = \langle h_1, h_2, \ldots, h_m : R\rangle$ be a presentation of H. Since H is normal in G, if $\bar{k}_i \in G$ is such that $\psi(\bar{k}_i) = k_i$ we have relations of the form

$$\bar{k}_r h_j^{\eta} \bar{k}_r^{-1} = w(r, j, \eta)$$

where $\eta = \pm 1$ and $w(r, j, \eta)$ is a word in the generators h_i and their inverses. Because H is a normal subgroup we also have the relations

$$\bar{k}_r \bar{k}_s = z(r, s)\bar{k}_{t(r,s)}$$

where $z(r, s)$ is a word in the generators h_i and their inverses determined by the relation $k_r k_s = k_{t(r,s)}$ in the multiplication table of K. So a presentation of G is

$$G = \langle \bar{k}_2, \ldots, \bar{k}_n, h_1, h_2, \ldots, h_m;\ \bar{k}_r^{-1} h_j^{\eta} \bar{k}_r = w(r, j, \eta),\ \bar{k}_r \bar{k}_s = z(r, s)\bar{k}_{t(r,s)}, R\rangle,$$

where $r, s = 2, \ldots, n, j = 1, 2, \ldots, m$, and $\eta = \pm 1$.

Let $\mathcal{M}$ be a pushdown automaton accepting the Word Problem of H for its presentation above. The idea of constructing a pushdown automaton $\widehat{\mathcal{M}}$ to accept the Word Problem of G for the above presentation is very simple. On reading a word w, the automaton $\widehat{\mathcal{M}}$ uses extra master control states to keep track of the image $\psi(w)$ in K and uses the stack to simulate $\mathcal{M}$ on the Word Problem of H. The automaton $\widehat{\mathcal{M}}$ starts with empty stack in the master control state corresponding to 1_K. If $\widehat{\mathcal{M}}$ is in the master control state corresponding to k_r and $\widehat{\mathcal{M}}$ reads a letter $\overline{k_s}$ it uses a sequence of auxiliary states to simulate $\mathcal{M}$ reading the word $z(r, s)$ and then changes to the master control state corresponding to k_t where $k_t = k_r k_s$ in K. If $\widehat{\mathcal{M}}$ reads a letter h_j^{η} while in the master control state corresponding to k_r in the quotient group it uses a series of auxiliary states to simulate $\mathcal{M}$ reading the word $w(r, j, \eta)$. Finally, $\widehat{\mathcal{M}}$ accepts by having empty stack and master control state corresponding to 1_K. $\square$

Definition 3.14. A group G is *virtually free* if G contains a free subgroup H of finite index in G.

Corollary 3.15. *A finitely generated virtually free group is context-free.*

Proof. Let G be a finitely generated virtually-free group and let $H \subset G$ be a free subgroup of finite index. Then H is finitely generated and, as seen in Example 3.10, context-free. By the "if" part of the previous proposition, we have that G is context-free as well. $\square$

Muller and Schupp [76] proved the following characterization of groups with context-free Word Problem.

Theorem 3.16 (*Muller–Schupp*)**.** *Let G be a finitely generated group. Then G is context-free if and only if G is virtually free.*

Remark 3.17. In [21] Ceccherini-Silberstein and Woess introduced and studied the concept of a *context-free pair of groups*. Such a pair (G, K) consists of a finitely generated group $G = \langle X; R\rangle$ together with a subgroup $K \subset G$ for which the language consisting of all words over $\Sigma^* = X \cup X^{-1}$ representing an element in K is context-free. (When K reduces to the identity element, this clearly specializes to the above definition of G to be a context-free group.) These investigations were extended by Woess in [98] who applied them to the study of random walk asymptotics yielding a complete proof of the local limit theorem for return probabilities on any context-free group.

3.7. Subgroups and embeddability

We briefly mention some applications of formal language theory to subgroups and embeddability.

Definition 3.18. Let $G = \langle X; R \rangle$ be a finitely generated group with group alphabet $\Sigma = X \cup X^{-1}$. Let $\psi \colon \Sigma^* \to G$ be the natural map. Let $S \subset G$ be a subset. An *enumeration* of S is a subset $L \subset \Sigma^*$ such that $\psi(L) = S$. Then one says that L is a *regular* (resp. *context-free*, resp. *computable*) enumeration provided that L is a regular (resp. context-free, resp. computably enumerable) language.

Anisimov and Seifert [3] proved in 1975 the following theorem.

Theorem 3.19 (*Anisimov–Seifert*)**.** *Let G be a finitely generated group and let $H \subset G$ be a subgroup of G. Then H is finitely generated if and only if H has a regular enumeration.*

Anisimov and Seifert also proved that context-free groups are finitely presentable, a fact used in the proof of the characterization theorem. The following more general result is due to Frougny, Sakarovitch, and Schupp [35].

Theorem 3.20 (*Frougny–Sakarovitch–Schupp*)**.** *Let G be a finitely generated group and let $N \subset G$ be a normal subgroup of G. Then N is finitely generated as a normal subgroup (that is, N equals the normal closure of a finite set of elements of G) if and only if N has a context-free enumeration.*

Definition 3.21. A *computably enumerable presentation* (also called a *recursive presentation*) is a group presentation $G = \langle X; R \rangle$ where the set X of generators is finite and the set R of defining relators is computably enumerable.

Recall that a group H is said to be *embeddable* into a group G provided there exists an injective homomorphism $\psi \colon H \to G$. The remarkable Higman Embedding Theorem [48] shows that the connection between group theory and computability is intrinsic.

Theorem 3.22 (*Higman*)**.** *A finitely generated group H is embeddable into some finitely presented group if and only if H admits a computably enumerable presentation.*

3.8. Basic groups and simple languages

We next consider a special subclass of deterministic context-free languages.

Definition 3.23. Let Σ be a finite alphabet. A language $L \subset \Sigma^*$ is called *simple* if it is accepted by a 1-state deterministic pushdown automaton which accepts by empty stack and is required to halt when it empties its stack.

The convention that the automaton accepting a simple language halts on empty stack makes a simple language L *prefix-free*, that is, if $w = uv \in L$ with u and v nontrivial then $u \notin L$. The main reference for simple languages is Harrison [46].

Recall that given a language $L \subset \Sigma^*$, the *Kleene star* of L is the language L^* over Σ defined by

$$L^* = \{w_1 w_2 \cdots w_n : w_i \in L \text{ where } i = 1, 2, \ldots, n \text{ and } n = 0, 1, 2, \ldots\}. \tag{3.3}$$

In other words, L^* is the submonoid of Σ^* generated by L.

Since we are interested in groups, the convention that the automaton must halt on empty stack is rather unnatural. Note that the language accepted by a 1-state deterministic pushdown automaton which is not required to halt on empty stack is the Kleene star L^*, of a simple language L (see Equation (3.3)).

Example 3.24. We show that the Word Problem for a finite group with respect to its multiplication table presentation is the Kleene star of a simple language. Let $G = \{g_1, g_2, \dots, g_n\}$ (with $g_1 = 1_G$) be a finite group and consider its multiplication table presentation

$$G = \langle g_2, \dots, g_n; g_i g_j = g_{k(i,j)} \rangle$$

(see Example 3.3.(a)). Let $\mathcal{M}$ be the deterministic single state pushdown automaton whose input alphabet and stack alphabet are the set $\{g_2, \dots, g_n\}$ of non-identity elements of G. The automaton $\mathcal{M}$ starts with empty stack and will always have at most one symbol on the stack. If the stack is empty and $\mathcal{M}$ reads g_i then $\mathcal{M}$ puts g_i on the stack. If the symbol on the stack is g_i and $\mathcal{M}$ reads g_j then $\mathcal{M}$ replaces g_i by the product $g_{k(i,j)}$ if $g_i g_j$ is not the identity of G and $\mathcal{M}$ empties the stack otherwise. It is clear that $\mathcal{M}$ has empty stack exactly when the product of the elements it has read so far is the identity, so $L(\mathcal{M})^* = \mathrm{WP}(G : g_2, \dots, g_n; g_i g_j = g_{k(i,j)})$.

Definition 3.25. A group G is called *basic* if it is the free product of finitely many finite groups and a free group of finite rank, i.e, $G \cong G_1 * G_2 * \cdots * G_k * F_n$, where G_i is a finite group, $i = 1, 2, \dots, k$, and F_n is the free group of rank n, with $k, n \geq 0$.

Note that finite groups and finitely generated free groups are basic groups. We saw in Example 3.24 that the Word Problem of a finite group with respect to the multiplication table presentation is the star of a simple language. Analogously, it follows from Example 3.10 that the Word Problem of a finitely generated free group with respect to the free presentation is the star of a simple language as well. More generally, if we take the "canonical presentation" of a basic group given by the disjoint union of the multiplication table presentations of the finite factors and the free presentation of the free group, then the corresponding Word Problem is the star of a simple language. In general, however, having a Word Problem which is the Kleene star of a simple language depends on the given presentation. We give an example below (Example 3.29).

Haring-Smith [45] characterized groups whose Word Problem is the star of a simple language.

Theorem 3.26 (*Haring-Smith*)**.** *A finitely generated group G is basic if and only if it has a finitely generated presentation $G = \langle X; R \rangle$ such that the corresponding Word Problem is the Kleene star of a simple language.*

Haring-Smith [45] also gave the following geometric characterization of basic groups.

Theorem 3.27 (*Haring-Smith*)**.** *A group G is basic if and only if G has a finitely generated presentation such that in the corresponding Cayley graph Γ the following holds: for every vertex $v \in V(\Gamma)$ there are only finitely many cycles through v.*

Indeed, the Word Problem for a given presentation is the star of a simple language if and only if its Cayley graph satisfies the above geometric condition.

Example 3.28 (*The Modular Group*)**.** A presentation of the modular group $G = \mathrm{PSL}(2, \mathbb{Z}) \cong (\mathbb{Z}/2\mathbb{Z}) * (\mathbb{Z}/3\mathbb{Z})$ is $G = \langle x, y; x^2, y^3 \rangle$. The corresponding Cayley graph Γ is represented in Fig. 10.

As illustrated in Fig. 10, for every vertex $v \in V(\Gamma)$ there are exactly two cycles through v, namely (e_1, e_2, e_3) and $({e_3}^{-1}, {e_2}^{-1}, {e_1}^{-1})$, where $e_1 = (v, y, vy)$, $e_2 = (vy, y, vy^2)$, and $e_3 = (vy^2, y, v)$. As usual, for an edge e we denote by e^{-1} the opposite edge (see the drawing convention for symmetric labelled graphs at page 12).

Example 3.29. Consider the presentation $\langle x, y; y = x^2 \rangle$ of the infinite cyclic group. In the Cayley graph of this presentation there are infinitely many cycles through a vertex (see Fig. 11) and the Word Problem for this presentation is not the star of a simple language.

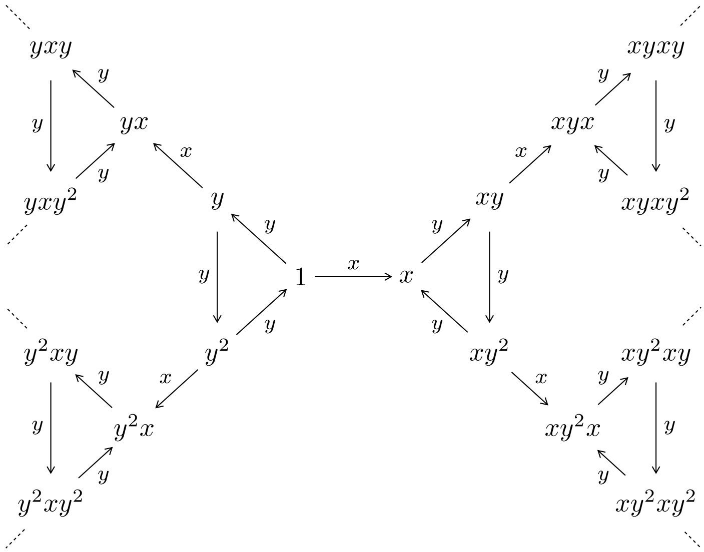

Fig. 10. The Cayley graph of the modular group $\mathbb{Z}/2\mathbb{Z} * \mathbb{Z}/3\mathbb{Z} = \langle x, y;\ x^2 = y^3 = 1\rangle$.

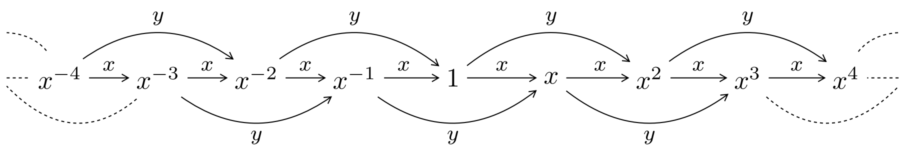

Fig. 11. The Cayley graph of the group $\mathbb{Z} = \langle x, y; y = x^2\rangle$.

4. Finitely generated graphs and ends

4.1. Finitely generated graphs

We need a framework in which we can discuss both Cayley graphs of finitely generated groups and complete transition graphs of pushdown automata. The following definition is from [77].

Definition 4.1. A *finitely generated graph* is a rooted labelled graph $\Gamma = (V, E, \Sigma, v_0)$ with a uniform upper bound on the degrees of vertices, and which is connected from v_0, that is, for every vertex $v \in V$, there is a directed path from v_0 to v.

The Cayley graph of a finitely generated group is clearly a finitely generated graph. Other examples of finitely generated graphs are provided by the complete transition graph of pushdown automata that we now define.

Definition 4.2 (*The Complete Transition Graph of a Pushdown Automaton*)**.** Let $\mathcal{M} = (Q, \Sigma, Z, \delta, q_0, F, z_0)$ be a pushdown automaton. The *complete transition graph* of $\mathcal{M}$ is the labelled graph $\Gamma = \Gamma(\mathcal{M})$ defined as follows. The initial vertex is the initial configuration (q_0, z_0). The vertex set V is the subset of $Q \times Z^*$ consisting of all configurations (q, ζ) which are reachable from the initial configuration on reading some possible input $w \in \Sigma^*$. In our previous notation,

$$V = \{(q, \zeta) : \mathcal{M} \overset{*}{\underset{w}{\vdash}} (q, \zeta),\ w \in \Sigma^*\}.$$

If $v = (q, \zeta)$ and $v' = (q', \zeta')$ are two vertices, then there is an oriented edge labelled by $a \in \Sigma$ from v to v' if and only if $\zeta = \zeta_0 z$ with $z \in Z$ such that there exists $(q', \zeta_1) \in \delta(q, a, z)$ satisfying $\zeta' = \zeta_0 \zeta_1$.

Note that $\Gamma(\mathcal{M})$ is connected from v_0 by definition and that there is an upper bound on the degrees of vertices. Thus the complete transition graph $\Gamma(\mathcal{M})$ of a pushdown automaton $\mathcal{M}$ is a finitely generated graph.

Example 4.3. Consider the deterministic pushdown automaton $\mathcal{M} = (Q, \Sigma, Z, \delta, q_0, F, z_0)$ with $Q = F = \{q_0\}$, $\Sigma = Z = \{0, 1\}$, $z_0 = \varepsilon$, and transition function defined by

$$\delta(q_0, a, z) = (q_0, za)$$

for all $a, z \in \{0, 1\}$. Then the associated complete transition graph of $\mathcal{M}$ is isomorphic to the rooted infinite binary tree T_2 (see Fig. 4).

4.2. Ends of finitely generated graphs

Let $\Gamma = (V, E, \Sigma, v_0)$ be a finitely generated graph. Intuitively, an *end* of Γ is a way to "go to infinity" in Γ. Although a finitely generated graph is a directed graph, in order to discuss ends, we need to consider undirected paths. Let Γ' be the graph obtained by considering Γ as an undirected graph. So if (u, σ, v) is an edge of Γ then both (u, v) and (v, u) are edges of Γ'. In short, one now ignores labels and the orientation of edges. An *undirected path* in Γ is a sequence of edges $(u_1, v_1), (v_1, v_2), \ldots, (v_i, v_{i+1}), \ldots, (v_n, v_{n+1})$ forming a path in Γ'.

For a non-negative integer n, we denote by Γ_n the subgraph of Γ whose vertex set V_n consists of all vertices $v \in V$ such that there exists an *undirected path* π with $|\pi| \leq n$ from the origin v_0 to v and whose edge set consists of the edges of Γ between two such vertices. Γ_n is called the *ball* of radius n centred at the basepoint v_0 of Γ.

Let n be a non-negative integer. It follows from the finiteness of the degrees of the vertices of Γ that there are only finitely many connected components of $\Gamma \setminus \Gamma_n$. Let us denote them by $\Gamma_{n,1}, \Gamma_{n,2}, \ldots, \Gamma_{n,k(n)}$. Let $e(n)$ be the number of *infinite* connected components of $\Gamma \setminus \Gamma_n$. Note that $0 \leq e(n) \leq k(n)$. Moreover, it is easy to see that $e(n)$ is a non-decreasing function of n. Thus the following limit exists in $\mathbb{R} \cup \{\infty\}$:

$$e(\Gamma) = \lim_{n \to \infty} e(n).$$

It is called the *number of ends* of Γ.

Example 4.4. (a) Let Γ be a finite graph and fix an arbitrary vertex $v_0 \in V(\Gamma)$. For every $n \geq 0$ one has $\Gamma \setminus \Gamma_n$ is finite and, in particular, has no infinite connected components, that is, $e(n) = 0$. It follows that $e(\Gamma) = 0$.

(b) Let $\Gamma = T_2$ be the rooted infinite binary tree. Then for every non-negative integer n, the vertex set of the ball of radius n centred at $v_0 = \varepsilon$ consists of all words in $\{0, 1\}^*$ having length at most n. Each connected component of $\Gamma \setminus \Gamma_n$ has vertex subset $V_w \subset V$ consisting of all words in $\{0, 1\}^*$ with proper prefix w, where $w \in \{0, 1\}^n$ is a word of length n. Since there are 2^n distinct words of length n over the alphabet $\{0, 1\}$, we have $e(n) = 2^n$ for all $n \geq 0$, so that $e(\Gamma) = \infty$.

(c) Let Γ be the Cayley graph of the infinite cyclic group $\mathbb{Z} = \langle x \rangle$. Then for every non-negative integer n the ball of radius n centred at $v_0 = 1_{\mathbb{Z}}$ is the "interval" from x^{-n} to x^n. Thus, $\Gamma \setminus \Gamma_n$ consists of the two disjoint intervals $C_{<n} = \{x^m : m < -n\}$ and $C_{>n} = \{x^m : m > n\}$. Thus $e(n) = 2$ for all $n \geq 1$ so that $e(\Gamma) = 2$.

(d) Let Γ be the Cayley graph of $\mathbb{Z}^2$ with respect to the presentation $\mathbb{Z}^2 = \langle x, y; [x, y] \rangle$. Then for every non-negative integer n the ball of radius n centred at the origin is the "square" $\Gamma_n = \{x^p y^q : |p| + |q| \leq n\}$. Thus, $\Gamma \setminus \Gamma_n$ consists of a single connected component, namely $C_{>n} = \{x^p y^q : |p| + |q| > n\}$. Hence $e(n) = 1$ for all $n \geq 0$ so that $e(\Gamma) = 1$.

Remark 4.5. If G is a finitely generated group then the number of ends of the Cayley graph of any finitely generated presentation of G is the same. Thus $e(G)$, the number of ends of G, is well defined and does not depend on the presentation. It is a fact that the number of ends of any finitely generated group is either 0, 1, 2, or ∞. We also remark that if $e(G) = \infty$ then G contains nonabelian free groups (see, e.g. [55,56,71,99]).

A very powerful result of Stallings [92] is the Stallings Structure Theorem.

Theorem 4.6 (*Stallings*). *Let G be a finitely generated group. Then $e(G) > 1$ if and only if one of the following holds:*

- *G admits a splitting $G = H *_C K$ as a free product with amalgamation, where C is a finite proper subgroup of both H and K;*
- *G admits a splitting $G = \langle H, t; tC_1t^{-1} = C_2\rangle$ as an HNN-extension, where C_1 and C_2 are isomorphic finite subgroups of H.*

The proof of the characterization of context-free groups as finitely generated virtually free groups depends heavily on the Stallings Structure Theorem. A consequence of the geometric characterization of context-free groups is that every finitely generated subgroup of a context-free group is either finite or has more than one end. This opens the way to a proof by induction but needs the notion of accessibility. A finitely generated group is *accessible* if the process of taking repeated splittings as in Stallings' theorem must halt after a finite number of steps. That is, one splits G as $H *_C K$ or as an HNN-extension $\langle H, t : tC_1t^{-1} = C_2\rangle$ according to the theorem and then splits H and K or just H in the HNN case, etc. Accessibility of context-free groups is needed to complete the characterization of context-free groups as virtually-free groups. (See Theorem 3.16.)

Senizergues [90] proved the following result (see also [91]).

Theorem 4.7 (*Senizergues*). *If G is a context-free group then there are only finitely many conjugacy classes of finite subgroups of G.*

Linnell [61] proved that any finitely generated group with only finitely many conjugacy classes of finite subgroups is accessible. In conjunction with Senizergues' theorem this shows that any context-free group is accessible. Dunwoody [30] later proved that *all* finitely presentable groups are accessible. Recall that Anisimov and Seifert proved that context-free groups are finitely presentable. (See the comments after Theorem 3.19.) Note that there exist finitely generated groups that are not accessible (see [31]).

4.3. Graphs with finitary end structure

We have seen that $e(\mathbb{Z}^2) = 1$ while $e(T_2) = \infty$. Later, in the section on monadic logic, we shall see that there is a precise sense in which, from the point of view of logical complexity, the Cayley graph of $\mathbb{Z}^2$ is infinitely more complicated than the rooted infinite binary tree T_2. So the *number* of ends is not a good measure of logical complexity but it turns out that we can still use ends to measure complexity.

Definition 4.8. Let Γ be a finitely generated graph. Denote by $c(\Gamma)$ the number of *end-isomorphism classes* of connected components of $\Gamma \setminus \Gamma_n$ over *all* components and all $n \geq 1$. An *end-isomorphism* between connected components C of $\Gamma \setminus \Gamma_n$ and C' of $\Gamma \setminus \Gamma_{n'}$ is a labelled graph isomorphism which additionally maps the points of Γ_n at distance n from v_0 to the points of $\Gamma_{n'}$ at distance n' from v_0 (thus respecting the end structure). Note that although we undirected the graph to define the connected components, we are using the directed structure of Γ to define end-isomorphisms.

Example 4.9 (*Compare with* Example 4.4).

(a) Let Γ be a finite graph. The number of all connected components of $\Gamma \setminus \Gamma_n$, $n \geq 1$, equals the number of all connected components of $\Gamma \setminus \Gamma_1, \Gamma \setminus \Gamma_2, \ldots, \Gamma \setminus \Gamma_{d-1}$, where $d = \max\{\mathrm{dist}(v, v_0) : v \in V(\Gamma)\}$, and is therefore finite. It follows that $c(\Gamma) < \infty$.

(b) Let $\Gamma = T_2$ be the rooted infinite binary tree, say with label 0 on left successor edges and label 1 on right successor edges. Then for every $n \in \mathbb{N}$ and every component C of $\Gamma \setminus \Gamma_n$ the graph C is a rooted infinite binary tree isomorphic to Γ. Thus $c(\Gamma) = 1$.
(c) Let Γ be the Cayley graph of $\mathbb{Z}$ with respect to the standard presentation. Recall that Γ is the infinite line (see Fig. 6) with a directed edge labelled by x from vertex x^n to vertex x^{n+1} for all $n \in \mathbb{Z}$. If we remove a ball Γ_r, $r \geq 1$, then there are always two components. Call these components the "left" component and the "right "component. These two components are not isomorphic as labelled graphs since edges with label x go from vertex x^n to vertex x^{n+1}. However, all right components are isomorphic to each other and all left components are isomorphic to each other. Thus $c(\Gamma) = 2$.
(d) Let Γ be the Cayley graph of $\mathbb{Z}^2$ with presentation $\langle x, y; [x, y]\rangle$ (see Fig. 8). Then, for every non-negative integer n the ball of radius n centred at the identity is the "square" $\Gamma_n = \{x^p y^q : |p|+|q| \leq n\}$. It is clear that the graphs $\Gamma \setminus \Gamma_n$ are pairwise non-isomorphic (look at the finite boundaries!) so that $c(\Gamma) = \infty$.

Definition 4.10. A finitely generated graph Γ has *finitary end-structure* if $c(\Gamma) < \infty$. A finitely generated graph is *context-free* if there exists a pushdown automaton $\mathcal{M}$ such that Γ is label-isomorphic to $\Gamma(\mathcal{M})$.

It turns out that there is a characterization of finitely generated graphs with finitary end-structure.

Theorem 4.11 (*Muller–Schupp*)**.** *Let Γ be a finitely generated graph. Then Γ has finitary end-structure if and only if Γ is context-free.*

The necessary condition of the theorem is the "easy part" while the sufficient condition is "hard". An analysis of the proof shows that finitely generated graphs Γ with $c(\Gamma) < \infty$ are "very treelike" (see also [100]). Indeed, Γ contains a rational subtree of finite index in the sense that there is a subtree T of Γ defined by a finite automaton such that every vertex of Γ is within a fixed distance from some vertex of T. Putting the characterization of graphs Γ with $c(\Gamma) < \infty$ together with the characterization of context-free groups we have the following result.

Corollary 4.12. *Let G be a finitely generated group and let Γ be the Cayley graph of any finitely generated presentation of G. Then $c(\Gamma) < \infty$ if and only if G is virtually free.*

5. Second-order monadic logic, the Domino Problem, and decidability

5.1. Second-order monadic logic and the theorems of Büchi and Rabin

The reader is probably familiar with *first-order logic* in which the quantifiers $\exists$ (there exists) and $\forall$ (for all) range only over individual elements of a given structure. The first-order language for a structure includes the quantifiers, variables $x, y, z, \ldots$ for individual elements and the Boolean connectives $\neg$ (negation), $\vee$ (or), and $\wedge$ (and). There are function and relation symbols for the operations and relations of the structure, including the relation of equality. For more on first-order logic see the monograph by Enderton [32].

Example 5.1 (*Group Axioms*)**.** The usual axioms which define a group are expressible in first-order logic. A quadruple $\langle G, *, ^{-1}, 1_G\rangle$, where G is a set with a binary function symbol $*$, a unary function symbol $^{-1}$, and a 0-ary constant symbol 1_G, defines a group provided that:

- $\forall x\, \forall y\, \forall z[(x * y) * z = x * (y * z)]$ (associative property);
- $\forall x[x * 1_G = 1_G * x = x]$ (existence of an identity element);
- $\forall x[x * x^{-1} = x^{-1} * x = 1_G]$ (existence of inverse elements).

In *monadic second-order* logic, one also has variables and quantifiers ranging over arbitrary subsets of the structure. The term "monadic" refers to the fact that we can quantify only over subsets of the given structure, and not over relations. Second-order logic with variables for arbitrary relations is sometimes called *full* second-order logic to distinguish it from the monadic version.

Example 5.2 (*Peano Axioms*)**.** Consider the language of *second-order Peano axioms* for *arithmetic* in which we have a unary function symbol s for the successor function, a constant symbol 0, the set membership symbol $\in$, the relation $\subseteq$ of set inclusion, and equality relation for both individual and set variables. The axioms are:

- $\forall x \neg[s(x) = 0]$
- $\forall y \exists x[y \neq 0 \Rightarrow y = s(x)]$
- $\forall x \forall y[s(x) = s(y) \Rightarrow x = y]$
- $\forall X[[0 \in X \wedge \forall x(x \in X \Rightarrow s(x) \in X)] \Rightarrow \forall y[y \in X]]$ (mathematical induction).

In standard second-order logic, these axioms define $\mathbb{N}$ with the successor function up to isomorphism. This theory is sometimes denoted by $S1S$, the *theory of one successor function*.

Büchi [11] introduced the theory of finite automata on infinite inputs to prove the following result.

Theorem 5.3 (*Büchi*)**.** *The monadic second-order theory $S1S$ is decidable.*

We next want to consider the monadic theory $S2S$ of two successor functions, that is, the monadic theory of the rooted infinite binary tree T_2. Individual variables and quantifiers can actually be eliminated since when a set has exactly one element is definable in the logic and we often adopt this point of view. Also, equality between sets is definable in terms of set inclusion. The set of vertices of the rooted infinite binary tree T_2 can be viewed as the set $\{0, 1\}^*$ of all finite words on $\{0, 1\}$. We have a constant for the root of the tree (which corresponds to the empty word ε) and two set-valued successor functions, 0 and 1. If S denotes a set of vertices then

$$S0 = \{v0 : v \in S\} \quad \text{and} \quad S1 = \{v1 : v \in S\}.$$

We also have the binary relation $\subseteq$ of set inclusion.

In 1969 Rabin [85] developed the theory of finite automata working on infinite trees and proved the following result.

Theorem 5.4 (*Rabin*)**.** *The monadic second-order theory $S2S$ is decidable.*

As a consequence of Rabin's theorem, the monadic second-order theory SnS of n successor functions is also decidable since it can be interpreted in $S2S$. Note that the above theories are about the geometry of the underlying graph. Analogously then, we can define the second-order monadic theory of any finitely generated graph $\Gamma = (V, E, \Sigma, v_0)$. We thus have again a constant for the origin of the graph v_0 and for each $a \in \Sigma$ we have a set-valued successor function where $Sa = \{v \in V : \exists u \in S$ such that $(u, a, v) \in E\}$ for all $S \subset V$.

5.2. The Domino Problem

Rabin's theorem is one of the most remarkable positive results on decidability. An important negative result is the unsolvability of the *Wang Domino Problem* in the plane. Whether or not it is possible to tile the plane with copies of a fixed finite set of square tiles with coloured edges was a question raised by Wang [96] in the late 1950s. Of course, when one places a tile next to another one, the colours on the matching edges must be the same. Wang showed that the *origin-constrained* problem is undecidable. In this version there is a fixed initial tile which must be used first. Indeed, fixing one tile is enough to show that one can directly simulate the Halting Problem for Turing machines in this context. Given a Turing machine $\mathcal{T}$ one can write down a set of tiles such that one can tile the entire plane if and only if $\mathcal{T}$ halts when started with a blank tape. The general Tiling Problem without an origin constraint was proved undecidable by Berger [7] in 1966. In 1971, Robinson [86] found a simpler proof of the undecidability of the general problem in the Euclidean plane.

This problem can be reformulated in terms of colouring vertices as follows. Let Γ be the Cayley graph of the standard presentation $\mathbb{Z}^2 = \langle x, y; [x, y]\rangle$ of the free abelian group of rank 2. Let $C = \{c_1, c_2, \ldots, c_k\}$ be a finite set of *colours*. The *standard neighbourhood* of a vertex v in Γ consists of v and its four neighbours: vx, vx^{-1}, vy, and vy^{-1} (see Fig. 12).

$$\begin{tikzcd}
 & vy & \\
vx^{-1} \arrow[r, "x"] & v \arrow[u, "y"] \arrow[r, "x"] & vx \\
 & vy^{-1} \arrow[u, "y"] &
\end{tikzcd}$$

Fig. 12. The standard neighbourhood of a vertex v in the Cayley graph of $\mathbb{Z}^2 = \langle x, y; [x, y]\rangle$.

We are also given a set $\mathcal{F}$ of *forbidden patterns* where a *pattern* $p \in C^5$ is a colouring of the vertices of the standard neighbourhood with colours from C. The *Domino Problem* for $\mathbb{Z}^2$ is the following decision problem: given a pair $(C, \mathcal{F})$ as above, can all the vertices of the Cayley graph Γ be coloured so that there are no forbidden patterns? Note that since Γ can be viewed as the dual graph of the tessellation by squares, this version is easily seen to be equivalent to the original formulation in terms of square tiles.

Our reformulation of the Domino Problem applies to an arbitrary finitely generated group G. Also, the Domino Problem is easily expressible in terms of the monadic second-order logic of the Cayley graph Γ of G with respect to the given presentation. A tuple $(C_1, C_2, \ldots, C_k)$ of sets of elements of G is a *disjoint cover of* G if every element of G belongs to exactly one of the C_i. (A disjoint cover differs from a partition only in that some of the C_i may be empty.) We need only say that there is a disjoint cover $(C_1, C_2, \ldots, C_k)$ of the vertices corresponding to the colours $c_1, c_2, \ldots, c_k$ such that there are no forbidden patterns. For example, if the i-th pattern in $\mathcal{F}$ centred at v has colour c_v at v and colours c_x, $c_{x^{-1}}$, c_y, and $c_{y^{-1}}$ at vx, vx^{-1}, vy, and vy^{-1} respectively, we abbreviate this as p_i, and we must say that such a pattern does not occur. We can write this as:

$$\exists C_1 \exists C_2 \cdots \exists C_k \forall v \left[\left[\bigvee_i [v \in C_i]\right] \wedge \left[\bigwedge_{i<j} [v \in C_i \Rightarrow v \notin C_j]\right] \wedge \left[\bigwedge_{p_i \in \mathcal{F}} \neg p_i\right]\right].$$

Note that from the point of view of logical complexity, measured in terms of alternation of quantifiers, the sentence above is very simple. It consists of one block of existential set quantifiers followed by one universal individual quantifier and such sentences are already undecidable. There is thus a precise sense in which the monadic logic of the Cayley graph of $\mathbb{Z}^2$ is infinitely more complicated than the monadic logic of the infinite binary tree, where the entire monadic theory is decidable.

Recently, Margenstern [66] (see also [67] for a shorter account) proved that the general Tiling Problem of the *hyperbolic plane* is undecidable by using a regular polygon as the basic shape of the tiles. Robinson raised this problem in the above mentioned paper and in 1978 he proved that the origin-constrained problem is undecidable for the hyperbolic plane [87]. The fundamental group of a closed orientable surface of genus 2 has a presentation $G_2 = \langle a, b, c, d; [a, b][c, d]\rangle$. The corresponding Cayley graph induces a tessellation of the hyperbolic plane by regular octagons and every vertex is on exactly eight such octagons (thus the graph is self-dual). We can reformulate Margenstern's undecidability result in group-theoretical language as follows.

Theorem 5.5 (*Margenstern*)**.** *The Domino Problem for the surface group G_2 is undecidable.*

5.3. Decidability of the monadic second-order theory for context-free groups

Recall that a finitely generated group G has context-free Word Problem if and only if G is virtually free (see Theorem 3.16). Now the Cayley graph of a finitely generated virtually free group has a regular tree of finite index. Namely, the subgraph corresponding to the Cayley graph of the free subgroup of finite index. In this case one can reduce the monadic theory of G to the monadic theory of the subtree. As a consequence, we have the following result [78].

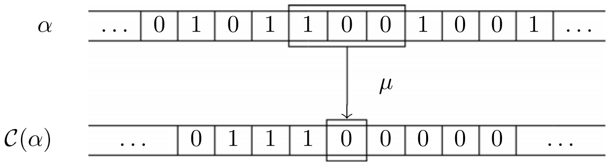

Fig. 13. The cellular automaton defined by the majority action on $\mathbb{Z}$.

Theorem 5.6 (*Muller–Schupp*)**.** *The monadic second-order theory of a Cayley graph of a context-free group is decidable.*

Corollary 5.7. *The Domino Problem for context-free groups is decidable.*

Kuske and Lohrey [59] have recently proved the converse to Theorem 5.6.

Theorem 5.8 (*Kuske–Lohrey*)**.** *If the monadic second-order theory of a Cayley graph of a finitely generated group is decidable, then the group is context-free.*

In the section on graphs with finitary end structure, we mentioned that all such graphs also have a regular subtree of finite index. Thus we have the following result from [78].

Theorem 5.9 (*Muller–Schupp*)**.** *Let Γ be the complete transition graph of a pushdown automaton. Then the monadic second-order theory of Γ is decidable.*

6. Cellular automata on groups

Cellular automata were introduced by von Neumann [12,94] who used them to describe theoretical models of self-reproducing machines. Although originally defined on the lattice of integer points in Euclidean plane, cellular automata can be defined over any group.

Let G be a group, called the *universe*, and let Σ be a finite alphabet called the set of *states* (or *colours*). Denote by Σ^G the set of all maps $\alpha\colon G \to \Sigma$, called *configurations*. When equipped with the prodiscrete topology, that is, the product topology obtained by taking the discrete topology on each factor Σ of $\Sigma^G = \prod_{g\in G} \Sigma$, the configuration space becomes a compact, Hausdorff, totally disconnected topological space. There is a natural continuous left action of G on Σ^G given by $g\alpha(h) = \alpha(g^{-1}h)$ for all $g, h \in G$ and $\alpha \in \Sigma^G$. This action is called the *G-shift* on Σ^G.

Definition 6.1. A map $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is called a *cellular automaton* provided there exists a finite subset $M \subset G$ and a map $\mu\colon \Sigma^M \to \Sigma$ such that

$$\mathcal{C}(\alpha)(g) = \mu((g^{-1}\alpha)|_M) \tag{6.1}$$

for all $\alpha \in \Sigma^G$ and $g \in G$, where $(\cdot)|_M$ denotes the restriction to M. The subset $M \subset G$ is called a *local neighbourhood* (or *memory set*) for $\mathcal{C}$ and μ is the associated *local defining map*.

Example 6.2 (*The Majority Action on* $\mathbb{Z}$)**.** Consider $G = \mathbb{Z}$, $\Sigma = \{0, 1\}$, $M = \{-1, 0, 1\}$ and $\mu\colon \Sigma^M \equiv \Sigma^3 \to \Sigma$ defined by

$$\mu(a_{-1}, a_0, a_1) = \begin{cases} 1 & \text{if } a_{-1} + a_0 + a_1 \geq 2 \\ 0 & \text{otherwise.} \end{cases}$$

Fig. 13 illustrates the behaviour of the corresponding cellular automaton $\mathcal{C}\colon \Sigma^{\mathbb{Z}} \to \Sigma^{\mathbb{Z}}$. Note that $\mathcal{C}$ is surjective but not injective.

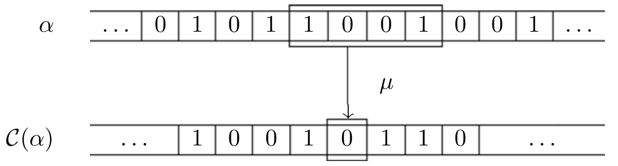

Fig. 14. The cellular automaton defined by the Hedlund marker.

Example 6.3 (*Hedlund's Marker [47]*)**.** Let $G = \mathbb{Z}$, $\Sigma = \{0, 1\}$, $M = \{-1, 0, 1, 2\}$ and $\mu\colon \Sigma^M \equiv \Sigma^4 \to \Sigma$ defined by

$$\mu(a_{-1}, a_0, a_1, a_2) = \begin{cases} 1 - a_0 & \text{if } (a_{-1}, a_1, a_2) = (0, 1, 0) \\ a_0 & \text{otherwise.} \end{cases}$$

The corresponding cellular automaton $\mathcal{C}\colon \Sigma^{\mathbb{Z}} \to \Sigma^{\mathbb{Z}}$ is a nontrivial involution of $\Sigma^{\mathbb{Z}}$. It is described in Fig. 14.

Example 6.4 (*Conway's Game of Life*)**.** Let $G = \mathbb{Z}^2$, $\Sigma = \{0, 1\}$, $M = \{-1, 0, 1\}^2 \subset \mathbb{Z}^2$ and $\mu\colon \Sigma^M \to \Sigma$ given by

$$\mu(y) = \begin{cases} 1 & \text{if } \begin{cases} \sum_{m \in M} y(m) = 3 \\ \text{or } \sum_{m \in M} y(m) = 4 \quad \text{and} \quad y((0,0)) = 1 \end{cases} \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$

for all $y \in \Sigma^M$. The corresponding cellular automaton $\mathcal{C}\colon \Sigma^{\mathbb{Z}^2} \to \Sigma^{\mathbb{Z}^2}$ describes the *Game of Life* due to Conway. One thinks of an element g of $G = \mathbb{Z}^2$ as a "cell" and the set gM (we use multiplicative notation) as the set consisting of its eight neighbouring cells, namely the North, North–East, East, South–East, South, South–West, West and North–West cells. We interpret state 0 as corresponding to the *absence* of life while state 1 corresponds to the *presence* of life. We thus refer to cells in state 0 as *dead* cells and to cells in state 1 as *live* cells. Finally, if $\alpha \in \Sigma^{\mathbb{Z}^2}$ is a configuration at time t, then $\mathcal{C}(\alpha)$ represents the evolution of the configuration at time $t + 1$. Then the cellular automaton in (6.2) evolves as follows.

- *Birth:* a cell that is dead at time t becomes alive at time $t + 1$ if and only if three of its neighbours are alive at time t.
- *Survival:* a cell that is alive at time t will remain alive at time $t + 1$ if and only if it has exactly two or three live neighbours at time t.
- *Death by loneliness:* a live cell that has at most one live neighbour at time t will be dead at time $t + 1$.
- *Death by overcrowding:* a cell that is alive at time t and has four or more live neighbours at time t, will be dead at time $t + 1$.

Fig. 15 illustrates all these cases. Note that $\mathcal{C}$ is not injective and it can be shown that $\mathcal{C}$ is not surjective either.

It easily follows from the definition that every cellular automaton $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is G-equivariant, i.e., $\mathcal{C}(g\alpha) = g\mathcal{C}(\alpha)$ for all $g \in G$ and $\alpha \in \Sigma^G$, and is continuous with respect to the prodiscrete topology on Σ^G. The Curtis–Hedlund–Lyndon Theorem ([47], [16, Theorem 1.8.1]) shows that the converse is also true.

It immediately follows from topological considerations and the Curtis–Hedlund–Lyndon Theorem that a bijective cellular automaton $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is *invertible*, in the sense that the inverse map $\mathcal{C}^{-1}\colon \Sigma^G \to \Sigma^G$ is also a cellular automaton.

$$\begin{array}{|c|c|c|}\hline 0&0&0\\\hline 0&0&0\\\hline 1&1&1\\\hline\end{array} \xrightarrow[\text{Birth}]{} \begin{array}{|c|c|c|}\hline &&\\\hline &1&\\\hline &&\\\hline\end{array} \qquad \begin{array}{|c|c|c|}\hline 0&0&0\\\hline 0&1&0\\\hline 1&1&*\\\hline\end{array} \xrightarrow[\text{Survival}]{} \begin{array}{|c|c|c|}\hline &&\\\hline &1&\\\hline &&\\\hline\end{array}$$

$$\begin{array}{|c|c|c|}\hline 0&0&0\\\hline 0&1&0\\\hline 0&0&*\\\hline\end{array} \xrightarrow[\text{Death}]{} \begin{array}{|c|c|c|}\hline &&\\\hline &0&\\\hline &&\\\hline\end{array} \qquad \begin{array}{|c|c|c|}\hline 1&*&*\\\hline 1&1&*\\\hline 1&1&*\\\hline\end{array} \xrightarrow[\text{Death}]{} \begin{array}{|c|c|c|}\hline &&\\\hline &0&\\\hline &&\\\hline\end{array}$$

Fig. 15. The evolution of a cell in the game of life. The symbol $*$ represents any symbol in $\{0,1\}$.

A map $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is called *pre-injective* (a terminology due to Gromov [43]) if whenever two configurations $\alpha, \beta \in \Sigma^G$ differ at only finitely many points (that is, the set $\{g \in G : \alpha(g) \neq \beta(g)\}$ is finite) and $\mathcal{C}(\alpha) = \mathcal{C}(\beta)$, then $\alpha = \beta$. Clearly pre-injectivity is a weaker form of injectivity.

Moore and Myhill [72,81] proved that for $G = \mathbb{Z}^d$, $d \geq 1$, a cellular automaton $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is surjective if and only if it is pre-injective. Necessity is due to Moore and sufficiency is due to Myhill. This result is often called the *Garden of Eden Theorem*. Regarding a cellular automaton as a dynamical system with discrete time, a configuration which is not in the image of the cellular automaton can only appear as an initial configuration, that is, at time $t = 0$. This motivates the biblical terminology. In 1993 Machì and Mignosi [63] extended the Garden of Eden theorem to finitely generated groups of subexponential growth (cf. the end of Section 3.2) and, finally, Ceccherini-Silberstein et al. [18] (see also [42]) further extended it to all amenable groups.

Recall that a group G is said to be *amenable*, a notion going back to von Neumann [95], if there exists a *left-invariant finitely additive probability measure* on G, that is, a map $m\colon \mathcal{P}(G) \to [0, 1]$ such that $m(G) = 1$, $m(A \cup B) = m(A) + m(B) - m(A \cap B)$ and $m(gA) = m(A)$, for all $A, B \in \mathcal{P}(G)$ and $g \in G$. Finite groups, abelian groups, and more generally solvable groups, groups of subexponential growth are amenable groups. On the other hand the free nonabelian groups are non-amenable.

Based on examples due to Muller [74], in [18] it is shown that if the group G contains a free nonabelian group (and is therefore non-amenable, since the class of amenable groups is closed under the operation of taking subgroups), then there exist examples of pre-injective (resp. surjective) cellular automata on G which are not surjective (resp. not pre-injective). Finally, Bartholdi in 2010 [4] (see also Theorem 5.12.1 in [16]) proved the converse to the amenable version of Moore's theorem in [18], namely that if every surjective cellular automaton $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is pre-injective, then the group G is amenable. This yields a new characterization of amenability in terms of cellular automata.

Following Gottschalk [38], we say that a group G is *surjunctive* provided that for every finite set Σ every injective cellular automaton $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ is surjective (and therefore bijective). It is an open problem to determine whether all groups are surjunctive or not. Lawton [60] (see also [16, Theorem 3.3.1]) showed that all residually finite groups (in particular, all virtually free groups) are surjunctive. Recall that a group is residually finite provided that the intersection of all its finite index subgroups reduces to the trivial group (see, e.g. [16, Chapter 2]). It immediately follows from the Garden of Eden Theorem for amenable groups that all amenable groups are surjunctive. Gromov [43] and Weiss [97] (see also [16, Theorem 7.8.1]) showed that all *sofic* groups are surjunctive. For the definition of soficity we refer to [16, Chapter 7]. We only mention that the class of sofic groups contains all residually finite groups and all amenable groups, and that it is not known if there are any non-sofic groups.

One is often interested in determining whether a cellular automaton is injective or surjective. In particular, the following question naturally arises: is it decidable, given a finite subset $M \subset G$ and a map $\mu\colon \Sigma^M \to \Sigma$, if the associated cellular automaton $\mathcal{C}\colon \Sigma^G \to \Sigma^G$ defined in (6.1) is surjective or not? Amoroso and Patt [1] proved in 1972 that if $G = \mathbb{Z}$ the above Surjectivity Problem is decidable. On the other hand, Kari [52–54] proved that the similar problem for cellular automata with finite alphabet over $\mathbb{Z}^d$, $d \geq 2$, is undecidable. His proof is based on Berger's undecidability result for the Domino Problem (see Section 5.2). It follows from the decidability of the monadic second-order theory of Cayley graphs of context-free groups (cf. Theorem 5.6) that the Surjectivity Problem for cellular automata defined over finitely generated virtually-free groups is decidable.

Indeed, that the cellular automaton is surjective is expressed by saying that for every disjoint cover $(C_1, C_2, \ldots, C_n)$ of G (where C_i represents the points currently in state $a_i \in \Sigma$) there is a disjoint cover

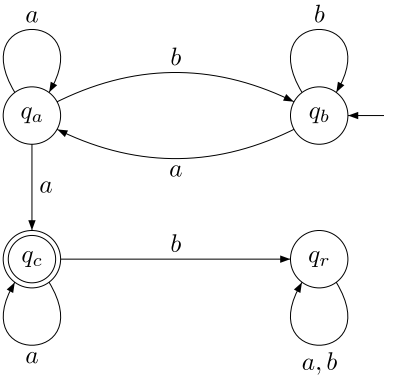

Fig. 16. The automaton accepting the infinite words $w \in \{a, b\}^{\mathbb{N}}$ containing only a finite number of b's.

$(P_1, P_2, \ldots, P_n)$ (the assignment of predecessor states) such that for every vertex v, one has $v \in C_i$ if and only if the points in the neighbourhood of v are in the correct P-sets for the local defining map μ to assign state a_i to v. This fact is easily expressible as a monadic second-order sentence. It similarly follows that the Injectivity and Bijectivity Problems are decidable for cellular automata on finitely generated virtually-free groups.

The following natural question is open.

Question. Are there any finitely generated groups which are not virtually free but for which the Surjectivity, Injectivity or Bijectivity Problems are decidable?

7. Finite automata on infinite inputs and infinite games of perfect information

7.1. Büchi acceptance and regular languages in $\Sigma^{\mathbb{N}}$

As mentioned in the Introduction, monadic sentences are too complicated to deal with directly. The theorems of Büchi (cf. Theorem 5.3) and of Rabin (cf. Theorem 5.4) are proved by developing a theory of finite automata working on infinite words and infinite trees respectively. Let $w = w_0 w_1 \cdots w_i w_{i+1} \cdots \in \Sigma^{\mathbb{N}}$ be an infinite word. (All our infinite words are infinite to the right.) In Büchi's original paper, a nondeterministic finite automaton working on a word $w \in \Sigma^{\mathbb{N}}$ is a tuple $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ exactly as in the case of automata on finite words (cf. Section 2.4). Thus, as usual, Q is a finite set of states, Σ is a finite alphabet, $q_0 \in Q$ is the initial state, $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ is the transition function and $F \subseteq Q$ is a set of final states. A *run* of $\mathcal{A}$ on w is a map $\rho : \mathbb{N} \to Q$ such that $\rho(0) = q_0$ and $\rho(i+1) \in \delta(\rho(i), w_i)$ for all $i \in \mathbb{N}$. We must now define when the automaton $\mathcal{A}$ *accepts* $w \in \Sigma^{\mathbb{N}}$, which we write as $\mathcal{A} \vdash w$. The definition of *Büchi acceptance* is that $\mathcal{A} \vdash w$ if there exists a run ρ of $\mathcal{A}$ on w such that some state from F occurs infinitely often. As in the case of finite words, we call the set

$$L(\mathcal{A}) = \{w \in \Sigma^{\mathbb{N}} : \mathcal{A} \vdash w\} \subset \Sigma^{\mathbb{N}}$$

the *language accepted* by $\mathcal{A}$. A subset $L \subseteq \Sigma^{\mathbb{N}}$ is a *regular language* if it is the language accepted by some finite automaton.

Example 7.1. Let $\Sigma = \{a, b\}$. We describe a finite automaton which accepts those infinite words $w \in \Sigma^{\mathbb{N}}$ containing b only a finite number of times. Let $\mathcal{A} = (Q, \Sigma, q_0, \delta, F)$ be a finite automaton where $Q = \{q_b, q_a, q_c, q_r\}$, $q_0 = q_b$, $F = \{q_c\}$ and

$$\delta(q_b, a) = q_a, \delta(q_b, b) = q_b,$$
$$\delta(q_a, a) = \{q_a, q_c\}, \delta(q_a, b) = q_b,$$
$$\delta(q_c, a) = q_c, \delta(q_c, b) = q_r,$$
$$\delta(q_r, a) = \delta(q_r, b) = q_r.$$

The automaton is illustrated in Fig. 16 and it works in the following way. When in state q_b, the automaton goes to q_a on reading a and remains in q_b on reading b. On reading a b in the state q_a it goes to state q_b. On reading an a in q_a the automaton can either remain in state q_a or "guess" that it will see no b's in the future by going to the "check" state q_c. In q_c the automaton remains in q_c as long as it sees only a's but goes to the reject state q_r if it ever reads a b. Once in q_r the automaton always remains in q_r on either input. Since $F = \{q_c\}$, in any accepting run the automaton must have guessed at some time that no more b's occur and must then always remain in q_c, thus seeing no more b's. And for any $w \in \Sigma^{\mathbb{N}}$ containing only finitely many b's there is an accepting run.

The overall goal is to associate with each monadic sentence ϕ of $S1S$ a finite automaton $\mathcal{A}_\phi$ such that ϕ is true if and only if $L(\mathcal{A}_\phi) \neq \varnothing$. In order to do this we need to establish the closure of regular languages under the three operations of union, complementation, and projection. These operations correspond to the logical connectives $\vee$, $\neg$, and $\exists$ respectively. If Σ and $\overline{\Sigma}$ are alphabets and $\pi : \Sigma \to \overline{\Sigma}$ is a map then π induces a function $\widehat{\pi} : \Sigma^{\mathbb{N}} \to \overline{\Sigma}^{\mathbb{N}}$ by letter-by-letter substitution. If $L \subset \Sigma^{\mathbb{N}}$ is a language, then $\widehat{\pi}(L) \subset \overline{\Sigma}^{\mathbb{N}}$ is the *projection* of L under π and we need to know that if L is a regular language over Σ then $\widehat{\pi}(L)$ is a regular language over $\overline{\Sigma}$.

The closure of regular languages with respect to the operation of union is easy to establish in essentially any model of finite automata. Also, projection is "easy" for nondeterministic automata, even on infinite words, and "hard" for deterministic automata. Suppose that $\pi : \Sigma \to \overline{\Sigma}$ is a function inducing the projection $\widehat{\pi} : \Sigma^{\omega} \to \overline{\Sigma}^{\omega}$ and that $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ is a nondeterministic automaton with alphabet Σ. To accept the projection of the language accepted by $\mathcal{A}$, we define a nondeterministic automaton $\widehat{\mathcal{A}}$ which, on reading a letter $\overline{a} \in \overline{\Sigma}$ can make any transition that $\mathcal{A}$ can make on any preimage of $\overline{a}$. Formally,

$$\widehat{\mathcal{A}} = (\mathcal{P}(Q), \overline{\Sigma}, \widehat{\delta}, \{q_0\}, \mathcal{P}(F)) \quad \text{where } \widehat{\delta}(S, \overline{a}) = \bigcup_{q \in S} \bigcup_{a \in \pi^{-1}(\overline{a})} \delta(q, a).$$

Note that even if we started with a deterministic automaton $\mathcal{A}$, the automaton $\widehat{\mathcal{A}}$ is nondeterministic.

7.2. Muller acceptance

In general, the closure of regular languages with respect to complementation is "hard" for nondeterministic automata, and regular languages in $\Sigma^{\mathbb{N}}$ recognized by using Büchi acceptance generally require using a nondeterministic automaton. The power of automata on infinite inputs is very sensitive to the acceptance condition used. Muller [75] introduced the concept of *Muller acceptance*, which is the most general type of acceptance commonly used.

Definition 7.2. A nondeterministic *Muller automaton* is a tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{F})$ where Q, Σ, δ and q_0 are exactly as for a nondeterministic finite automaton but $\mathcal{F} \subset \mathcal{P}(Q)$. Let $w \in \Sigma^{\mathbb{N}}$ be a word. If ρ is a run of $\mathcal{A}$ on w then we denote by $\mathrm{Inf}(\rho)$ the set of states occurring infinitely often in ρ. Then $\mathcal{A}$ *accepts* w if there exists a run ρ of $\mathcal{A}$ on w such that $\mathrm{Inf}(\rho) \in \mathcal{F}$.

Remark 7.3. If we compare Büchi acceptance with Muller acceptance, we have that the set of final states $F \subset S$ is now replaced by the "accepting" family $\mathcal{F}$. Moreover $w \in \Sigma^{\mathbb{N}}$ is Büchi-accepted if $\mathrm{Inf}(\rho) \cap F \neq \varnothing$, while it is Muller-accepted if $\mathrm{Inf}(\rho) \in \mathcal{F}$.

The following result was conjectured by Muller and then proved by McNaughton [70].

Theorem 7.4 (*McNaughton*)**.** *For any nondeterministic automaton on infinite words using Muller acceptance, there is an equivalent deterministic automaton using Muller acceptance.*

While the negation of a Büchi acceptance condition is not a Büchi condition, the negation of a Muller acceptance condition $\mathcal{F}$ is again a condition of the same type, namely the Muller condition

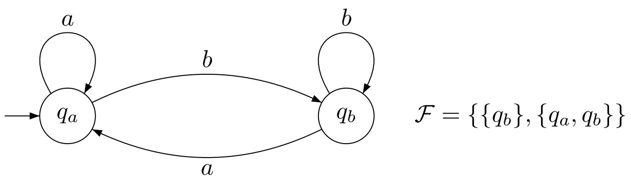

Fig. 17. The automaton accepting the infinite words $w \in \{a, b\}^{\mathbb{N}}$ containing an infinite number of b's.

defined by the accepting family $\mathcal{P}(Q) \setminus \mathcal{F}$. For a deterministic automaton $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{F})$ using Muller acceptance to accept the language $L(\mathcal{A})$ we have

$$\Sigma^* \setminus L(\mathcal{A}) = L(\neg\mathcal{A}) \quad \text{where } \neg\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{P}(Q) \setminus \mathcal{F}).$$

In short, $\neg\mathcal{A}$ is obtained from $\mathcal{A}$ by simply complementing the accepting family.

McNaughton's theorem thus proves that the class of regular languages of infinite words is closed under complementation. Proving McNaughton's theorem from scratch is not easy and it is an accident that determinizing the nondeterministic automaton of Example 7.1 is easy.

Example 7.5. Let $\Sigma = \{a, b\}$. We now present a deterministic finite automaton $\mathcal{A}$ using Muller acceptance which accepts exactly those words $w \in \Sigma^{\mathbb{N}}$ containing b infinitely often. Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{F})$ be the finite automaton in which $Q = \{q_a, q_b\}$, $\Sigma = \{a, b\}$, $q_0 = q_a$, $\mathcal{F} = \{\{q_b\}, \{q_a, q_b\}\}$ and

$$\delta(q_a, a) = q_a, \qquad \delta(q_a, b) = q_b,$$
$$\delta(q_b, a) = q_a, \qquad \delta(q_b, b) = q_b.$$

The automaton is illustrated in Fig. 17 and it works in the following way. The states q_a and q_b record which letter has just been read. On a word $w \in \Sigma^{\mathbb{N}}$ containing b infinitely often the set of states occurring infinitely often must be exactly $\{q_a, q_b\}$ in the case that both letters occur infinitely often or $\{q_b\}$ in the case that only b occurs infinitely often. Since $\mathcal{F}$ consists of these two sets, the automaton accepts exactly the desired words. Note that $\neg\mathcal{A} = (Q, \Sigma, \delta, q_0, \{\{q_a\}\})$ is a deterministic automaton using Muller acceptance which accepts exactly those words containing b only finitely many times (cf. Example 7.1).

Deciding the Emptiness Problem for non-deterministic Muller automata is easy. Given $\mathcal{A}$ with underlying graph Γ, the language $L(\mathcal{A}) \neq \varnothing$ if and only if there is a path in Γ from the initial state to a cycle containing exactly the states in some set $S \in \mathcal{F}$.

7.3. Rabin's theory

We now turn to considering automata on the infinite binary tree T_2. Recall that each vertex of T_2 is described by a finite word over the set $\{0, 1\}$ of the two possible directions. For a nondeterministic automaton with alphabet Σ working on T_2, a possible input α consists of an element $\alpha \in \Sigma^{T_2}$ which can be described as a copy of T_2 with all vertices labelled from Σ. In Rabin's model, a nondeterministic automaton is a 5-tuple $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{F})$, where Q, Σ, q_0 and $\mathcal{F}$ are defined as in Section 7.2. The transition function is of the form $\delta : Q \times \Sigma \to \mathcal{P}(Q \times Q)$. The automaton starts at the root ε in the initial state q_0. A copy of the automaton at a vertex v always sends one copy to the left successor of v and one copy to the right successor of v.

Example 7.6. If one has

$$\delta(q_0, a) = \{(q_1, q_3), (q_2, q_0)\},$$

then when the automaton is in state q_0 reading the letter a, it can send one copy to the left in state q_1 *and* one copy to the right in state q_3, *or* it can send one copy to the left in state q_2 *and* one copy to the right in state q_0. Note that both "and" and "or" occur in the description of the transition function. This situation is illustrated in Fig. 18.

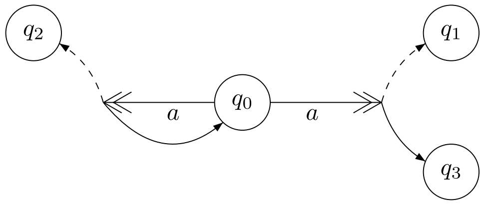

Fig. 18. An instance of the transition function in a Rabin automaton. The drawing convention is that the broken line visualizes the copy to the left, while the full line visualizes the copy to the right.

We must now define what it means for an automaton $\mathcal{A}$ to accept an input α, for which we write $\mathcal{A} \vdash \alpha$ as usual. An infinite path π through T_2 is a path starting at the origin ε such that each vertex in π has exactly one successor in π. Note that $\pi \in \{0, 1\}^{\mathbb{N}}$ and there are thus uncountably many distinct infinite paths through the tree. A *run* ρ of $\mathcal{A}$ on α is an element in Q^{T_2}, that is, a labelling of T_2 by states from Q such that for each vertex $v \in T_2$ we have

$$(\rho(v0), \rho(v1)) \in \delta(\rho(v), \alpha(v)).$$

We will again use Muller acceptance although Rabin used a different but equivalent condition. So we specify a family $\mathcal{F} \subseteq \mathcal{P}(Q)$. Given a run ρ and a path π, we define $\mathrm{Inf}(\rho, \pi)$ to be the set of states in ρ which occur infinitely often along the path π. Finally,

$$\mathcal{A} \vdash \alpha \quad \text{if } \exists \rho\, \forall \pi [\mathrm{Inf}(\rho\,|_\pi) \in \mathcal{F}].$$

In short, for every path π the set of states occurring infinitely often along π must be some set S in the accepting family $\mathcal{F}$. Note that S can vary with different paths.

Example 7.7. We extend Example 7.1. Suppose again that $\Sigma = \{a, b\}$ and we now want an automaton which accepts $\alpha \in \Sigma^{T_2}$ exactly if α contains some infinite path π on which b occurs only finitely often. Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{F})$ where $Q = \{q_a, q_b, q_d\}$, $q_0 = q_a$, $\mathcal{F} = \{\{q_a\}, \{q_d\}\}$, and

$$\begin{aligned} &\delta(q_a, a) = \{(q_d, q_a), (q_a, q_d)\}, && \delta(q_a, b) = \{(q_d, q_b), (q_b, q_d)\},\\ &\delta(q_b, a) = \{(q_d, q_a), (q_a, q_d)\}, && \delta(q_b, b) = \{(q_d, q_b), (q_b, q_d)\},\\ &\delta(q_d, a) = \delta(q_d, b) = \{(q_d, q_d)\}. \end{aligned}$$

The automaton is illustrated in Fig. 19 and works in the following way. Its overall strategy is to make a nondeterministic choice of the path π. On reading an a in state q_a, the automaton sends a copy in the "don't care" state q_d in one direction and a copy in q_a in the other direction. On reading a b in state q_a, the automaton sends a copy in the "don't care" state q_d in one direction and a copy in q_b the other direction. The state q_b functions similarly. If the automaton is in the "don't care" state q_d, it is not on the chosen path and so sends copies in q_d in both directions on reading either letter. It is easy to see that $\mathcal{A} \vdash \alpha$ if and only if α does contain an infinite path with only finitely many b's.

7.4. *Infinite games of perfect information*

Deterministic automata on trees are not very powerful and nondeterminism is essential. Rabin's proof of the closure of regular languages under complementation was very difficult. We now know that the best way to understand automata on infinite inputs is in terms of infinite games of perfect information, as introduced by Gale and Stewart [36].

Let Σ be a finite alphabet, let $\Sigma^{\mathbb{N}}$ denote the set of all infinite words over Σ, and let $\mathcal{W}$ be a subset of $\Sigma^{\mathbb{N}}$. We consider the following game between Player I and Player II. Player I chooses a letter $\sigma_1 \in \Sigma$ and Player II then chooses a letter $\sigma_2 \in \Sigma$. Continuing indefinitely, at step n Player I chooses a letter $\sigma_{2n-1} \in \Sigma$ and Player II then chooses a letter $\sigma_{2n} \in \Sigma$. The sequence of choices defines an infinite word $w = \sigma_1\sigma_2 \cdots \sigma_n \cdots \in \Sigma^{\mathbb{N}}$. Player I wins the game if $w \in \mathcal{W}$ and Player II wins otherwise. The basic question about such games is whether or not one of the players has a winning strategy,

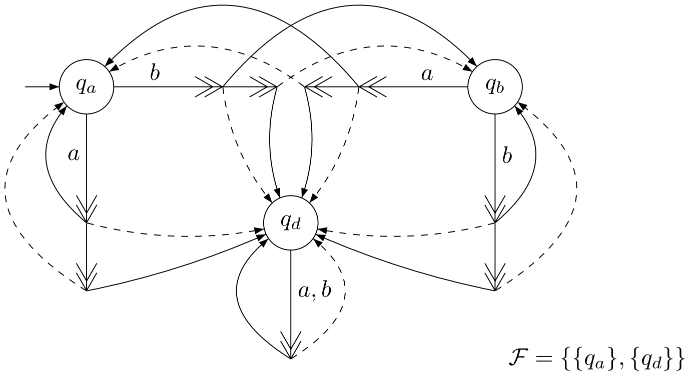

Fig. 19. The Rabin automaton defined in Example 7.7.

that is, a function $\phi : \Sigma^* \to \Sigma$ such that when a finite word u has already been played, the player then plays $\phi(u) \in \Sigma$ and always wins. Using the Axiom of Choice, it is possible to construct winning sets such that neither player has a winning strategy, but this cannot happen if the set $\mathcal{W}$ is not "too complicated".

Example 7.8. We show that if the set $\mathcal{W}$ is countable and $|\Sigma| \geq 2$ then the second player has a winning strategy by applying Cantor's diagonal argument. Let $w_i = w_{i,1}w_{i,2}\cdots w_{i,n}\cdots$ be the i-th word in $\mathcal{W}$. On his turn, play $2k$, Player II simply plays a letter different from $w_{2k,2k}$. Thus the word resulting from the set of plays is not in $\mathcal{W}$. Note that this simple example shows that strategies need not at all be effectively computable. Since the w_i are infinite words, even a single such word need not be computable since $\mathcal{W}$ is an arbitrary countable subset of $\Sigma^{\mathbb{N}}$.

The set $\Sigma^{\mathbb{N}}$ becomes a complete metric space by defining $\mathrm{dist}(v, w) = 2^{-j}$ for all $v = v_1v_2\cdots$ and $w = w_1w_2\cdots$, where j is the least index such that $w_j \neq v_j$. An important theorem of Martin [68,69] (see also [57, Sect. 20] and [73, Sect. 6F]) shows that if the set $\mathcal{W}$ is a Borel set then one of the two players must have a winning strategy. In applying infinite games to automata, one needs only consider winning conditions which are $F_{\delta,\sigma}$ and that such games are determined was proven by Davis [25] before Martin's general result. Given an automaton $\mathcal{A}$ and an input α, one defines the *acceptance game* $\mathcal{G}(\mathcal{A}, t)$ for $\mathcal{A}$ on the input α. The first player wins if $\mathcal{A}$ accepts α while the second player wins if $\mathcal{A}$ rejects.

Muller and Schupp [79] defined *alternating tree automata* as a generalization of nondeterministic automata working on trees. In this model, the transition function has the form $\delta : Q \times \Sigma \to \mathcal{L}(Q \times \{0, 1\})$, where $\mathcal{L}(Q \times \{0, 1\})$ is the free distributive lattice generated by all possible pairs (state, direction).

Example 7.9. We consider again a nondeterministic automaton in which

$$\delta(q_0, a) = \{(q_1, q_3), (q_2, q_0)\}$$

as in Example 7.6. In the lattice notation we can write this as

$$\delta(q_0, a) = [(q_1, 0) \wedge (q_3, 1)] \vee [(q_2, 0) \wedge (q_0, 1)].$$

Here the symbol $\vee$ stands for nondeterministic choice and $\wedge$ means "do both things".

We *dualize* a transition function of an alternating tree automaton by interchanging $\wedge$ and $\vee$ as usual. For the example above we have:

$$\widetilde{\delta}(q_0, a) = [(q_1, 0) \vee (q_3, 1)] \wedge [(q_2, 0) \vee (q_0, 1)].$$

Converting this expression to disjunctive normal form we have:

$$\widetilde{\delta}(q_0, a) = [(q_1, 0) \wedge (q_2, 0)] \vee [(q_1, 0) \wedge (q_0, 1)] \vee [(q_3, 1) \wedge (q_2, 0)] \vee [(q_3, 1) \wedge (q_0, 1)].$$

We interpret this as saying that when the automaton is in state q_0 reading the letter a it has a choice of sending one copy to the left in q_1 *and* another copy to the left in q_2, *or* sending a copy to the left in q_1 *and* a copy to the right in q_0, *or* a copy to the right in q_3 *and* a copy to the left in q_2, *or*, finally, a copy to the right in q_3 *and* another copy to the right in q_0. This is not a nondeterministic automaton but it is a perfectly good alternating automaton. Note that the automaton can send multiple copies in the same direction and is not required to send copies in all directions. It must, of course, send at least one copy in some direction.

We now have a framework general enough to always be able to dualize.

Definition 7.10. Let $\mathcal{A} = (Q, \Sigma, \delta, q_0, \mathcal{F})$ be an alternating automaton on the rooted infinite binary tree. Then the *dual automaton* of $\mathcal{A}$ is

$$\widetilde{\mathcal{A}} = (Q, \Sigma, \widetilde{\delta}, q_0, \widetilde{\mathcal{F}})$$

where $\widetilde{\delta}$ is obtained by dualizing the transition function δ, and the accepting family is $\widetilde{\mathcal{F}} = \mathcal{P}(Q) \setminus \mathcal{F}$.

It is clear from the definition that the dual of $\widetilde{\mathcal{A}}$ is just $\mathcal{A}$. One must carefully define the acceptance game $\mathcal{G}(\mathcal{A}, t)$ of $\mathcal{A}$ on an input α (for details see [79]). That this game is determined follows from Davis' theorem. In the alternating framework, it is easy to check that a winning strategy for the second player in $\mathcal{G}(\mathcal{A}, t)$ is a winning strategy for the first player in the acceptance game $\mathcal{G}(\widetilde{\mathcal{A}}, t)$ for the dual automaton. Thus complementation is easy for alternating automata and the following theorem is a consequence of pure determinacy.

Theorem 7.11 (*The Complementation Theorem*)**.** *If $\mathcal{A}$ is an alternating tree automaton accepting the language $L(\mathcal{A})$ then the dual automaton $\widetilde{\mathcal{A}}$ accepts the complementary language $\neg L(\mathcal{A})$.*

Of course, something must be hard for alternating automata and it is the operation of projection. The argument for nondeterministic automaton fails completely because there may be multiple copies of the automaton at the same vertex of the tree. So we must prove that given an alternating automaton, there is a nondeterministic automaton accepting the same language. Gurevich and Harrington [44] made a fundamental contribution to understanding automata on infinite inputs by showing that a winning strategy in the acceptance game for a nondeterministic automaton depends only on a finite amount of memory called the *later appearance record*. This is called the *Forgetful Determinacy Theorem* (see [44,101]). Muller and Schupp [79,80] used the later appearance record to prove the *Simulation Theorem* which states that there is an effective construction which, given an alternating automaton, produces a nondeterministic automaton accepting the same language.

Given the Complementation and Simulation theorems, most results have short conceptual proofs. As an illustration, we present a proof of McNaughton's theorem.

Proof of Theorem 7.4. There is a natural notion of an automaton which is alternating but still deterministic. Namely, one with no $\vee$'s in its transition function. The Simulation Theorem shows that if we start with a deterministic alternating automaton, then the simulating ordinary automaton is a deterministic automaton. If $\mathcal{A}$ is a nondeterministic automaton on the line (i.e. $|D| = 1$), using Muller acceptance, then $\mathcal{A}$ has only $\vee$'s in its transition function. Then its dual automaton $\widetilde{\mathcal{A}}$ has only $\wedge$'s in its transition function and therefore is a deterministic alternating automaton. By the Simulation Theorem we can construct a deterministic automaton $\mathcal{A}'$ on the line which accepts the same language L' as $\widetilde{\mathcal{A}}$. By the Complementation Theorem, L' is the complement of the language L accepted by $\mathcal{A}$. Since $\mathcal{A}'$ is deterministic we obtain a deterministic automaton $\neg\mathcal{A}'$ accepting the complement of L', that is, L, by simply complementing the accepting family of $\mathcal{A}$, thus establishing McNaughton's Theorem. □

Acknowledgements

We would like to express our deepest gratitude to Tony Martin for a very interesting and stimulating conversation on infinite games of perfect information and the Borel determinacy. Also, we wish to warmly thank Maurice Margenstern for providing fruitful information on the Domino Problem on the hyperbolic plane. Last but not least, we thank Ilya Kapovich for very helpful suggestions.

References

[1] S. Amoroso, Y.N. Patt, Decision procedures for surjectivity and injectivity of parallel maps for tessellation structures, J. Comput. System Sci. 6 (1972) 448–464.
[2] A.V. Anisimov, Über Gruppen-Sprachen, Kibernetika 4 (1971) 18–24.
[3] A.V. Anisimov, F.D. Seifert, Zur algebraischen Charakteristik der durch kontext-freie Sprachen definierten Gruppen, Elektron. Inform.-verarb. Kybernetik 11 (1975) 695–702.
[4] L. Bartholdi, Gardens of Eden and amenability on cellular automata, J. Eur. Math. Soc. 12 (2010) 241–248.
[5] L. Bartholdi, R.I. Grigorchuk, V.V. Nekrashevych, in: P.M. Grabner, W. Woess (Eds.), From Fractal Groups to Fractal Sets, in Fractals in Graz 2001, in: Trends in Mathematics, Birkhauser, 2003, pp. 25–118.
[6] L. Bartholdi, R.I. Grigorchuk, Z. Sunik, in: M. Hazenwinkel (Ed.), Branch Groups, in: Handbook of Algebra, vol. 3, Elsevier BV, 2003, pp. 989–1112.
[7] R. Berger, Undecidability of the Domino Problem, Mem. Amer. Math. Soc. 66 (1966).
[8] W.W. Boone, The Word Problem, Proc. Natl. Acad. Sci. USA 44 (1958) 1061–1065.
[9] M.R. Bridson, R.H. Gilman, Context-free languages of sub-exponential growth, J. Comput. System Sci. 64 (2002) 308–310.
[10] M.R. Bridson, A. Haefliger, Metric spaces of non-positive curvature, in: Grundlehren der Mathematischen Wissenschaften, vol. 319, Springer-Verlag, Berlin, 1999.
[11] J.R. Büchi, On a decision method in restricted second-order arithmetic, in: Proc. Logic, Methodology and Philosophy of Science, Proc. 1960 Int. Congr., 1962, pp. 1–11.
[12] A.W. Burks, von Neumann's self-reproducing automata, in: Essays on Cellular Automata, 1970, pp. 3–64.
[13] A. Cayley, Desiderata and suggestions. No. 1: the theory of groups. No. 2: graphical representation, Amer. J. Math. 1 (1878) 50–52. 174–176.
[14] T. Ceccherini-Silberstein, On the growth of linear languages, Adv. Appl. Math. 35 (2005) 243–253.
[15] T. Ceccherini-Silberstein, Growth and ergodicity of context-free languages II: the linear case, Trans. Amer. Math. Soc. 359 (2007) 605–618.
[16] T. Ceccherini-Silberstein, M. Coornaert, Cellular automata and groups, in: Springer Monographs in Mathematics, Springer-Verlag, Berlin, 2010.
[17] T. Ceccherini-Silberstein, A. Machì, F. Scarabotti, On the entropy of regular languages, Theoret. Comput. Sci. 307 (2003) 93–102.
[18] T. Ceccherini-Silberstein, A. Machì, F. Scarabotti, Amenable groups and cellular automata, Ann. Inst. Fourier 49 (1999) 673–685.
[19] T. Ceccherini-Silberstein, W. Woess, Growth and ergodicity of context-free languages, Trans. Amer. Math. Soc. 354 (2002) 459–4625.
[20] T. Ceccherini-Silberstein, W. Woess, Growth-sensitivity of context-free languages, Theoret. Comput. Sci. 307 (2003) 103–116.
[21] T. Ceccherini-Silberstein, W. Woess, Context-free pairs of groups I: Context-free pairs and graphs, European J. Combin. (2012). http://dx.doi.org/10.1016/j.ejc.2012.03.011.
[22] I. Chiswell, A Course in Formal Languages, Automata and Groups, Universitext. Springer-Verlag London, Ltd., London, 2009.
[23] N. Chomsky, Three models for the description of language, IRE Trans. Inform. Theory 2 (1956) 113–124.
[24] S.B. Cooper, Computability Theory, Chapman-Hall/CRC Mathematics, Boca Raton, 2004.
[25] M. Davis, Infinite games of perfect information, Ann. Math. Studies 52 (1964) 85–101.
[26] M. Dehn, Über unendliche diskontinuierliche Gruppen, Math. Annalen 71 (1911) 116–144.
[27] M. Dehn, Transformation der Kurven auf zweiseitigen Flächen, Math. Annalen 72 (1912) 413–421.
[28] P. de la Harpe, Topics in Geometric Group Theory, in: Chicago Lecture Notes in Mathematics, University of Chicago Press, Chicago, IL, 2000.
[29] P. de la Harpe, Topologie, théorie des groupes et problèmes de décision, Gaz. Math., Soc. Math. Fr. 125 (2010) 41–75.
[30] M.J. Dunwoody, The accessibility of finitely presented groups, Invent. Math. 81 (1985) 449–457.
[31] M.J. Dunwoody, An inaccessible group. in: Geometric group theory. Volume 1. Proceedings of the Symposium Held at the Sussex University, Brighton, UK, 1991. in: Lond. Math. Soc. Lect. Note Ser. vol. 181, 1993, pp. 75–78.
[32] H.B. Enderton, A Mathematical Introduction to Logic, Academic Press, New York, 1972.
[33] D.B.A. Epstein, J.W. Cannon, D.F. Hold, S.V.F. Levy, M.S. Paterson, W.P. Thurston, Word Processing in Groups, Jones and Barlett, 1992.
[34] H. Freudenthal, Über die Enden topologischer Räume und Gruppen, Math. Z. 33 (1931) 692–713.
[35] C. Frougny, J. Sakarovitch, P.E. Schupp, Finiteness conditions on subgroups and formal language theory, Proc. Lond. Math. Soc. 58 (1989) 74–88.
[36] D. Gale, F.M. Stewart, Infinite games with perfect information. In contrib. theory of games, II, Ann. Math. Stud. 28 (1953).
[37] S.M. Gersten, Dehn functions and ℓ_1-norms of finite presentations, in: Algorithms and Classification in Combinatorial Group Theory, Berkeley, CA, 1989, in: Math. Sci. Res. Inst. Publ., vol. 23, Springer, New York, 1992, pp. 195–224.
[38] W.H. Gottschalk, Some general dynamical notions. In recent advances in topological dynamics, Lect. Notes Math. 318 (1973) 120–125.

[39] R.I. Grigorchuk, Solved and unsolved problems around one group, in: L. Bartholdi, T. Ceccherini-Silberstein, T. Smirnova-Nagnibeda, A. Zuk (Eds.), Infinite Groups: Geometric, Combinatorial and Dynamical Aspects, in: Progress in Mathematics Series, vol. 248, 2005.
[40] R.I. Grigorchuk, A. Machì, An example of an indexed language of intermediate growth, Theoret. Comput. Sci. 215 (1999) 325–327.
[41] R.I. Grigorchuk, V.V. Nekrashevych, V.I. Sushshanskii, Automata, dynamical systems and infinite groups, Proc. Steklov Inst. Math. 231 (2000) 134–214.
[42] M. Gromov, Hyperbolic groups, in: Essays in Group Theory, in: Math. Sci. Res. Inst. Publ., vol. 8, Springer, New York, 1987, pp. 75–263.
[43] M. Gromov, Endomorphisms of symbolic algebraic varieties, J. Eur. Math. Soc. 1 (1999) 109–197.
[44] Y. Gurevich, L. Harrington, Trees, automata and games, in: Proc. 14-th ACM Symposium on the Theory of Computing, 1982, pp. 60–65.
[45] R.H. Haring-Smith, Groups and simple languages, Trans. Amer. Math. Soc. 279 (1983) 337–356.
[46] M.A. Harrison, Introduction to Formal Language Theory, Addison-Wesley, Reading, MA, 1978.
[47] G.A. Hedlund, Endomorphisms and automorphisms of the shift dynamical system, Math. Syst. Theory 3 (1969) 320–375.
[48] G. Higman, Subgroups of finitely presented groups, Proc. R. Soc. Ser. A. Math. Phys. Sci. 262 (1961) 455–475.
[49] J.E. Hopcroft, J.D. Ullman, Introduction to Automata Theory, Languages and Computation, Addison-Wesley, Reading, MA, 1979.
[50] H. Hopf, Enden offener Raüme und unendliche disdontinuierliche Gruppen, Comment. Math. Helv. 16 (1944) 81–100.
[51] R. Incitti, The growth function of context-free languages, Theoret. Comput. Sci. 255 (1999) 601–605.
[52] J. Kari, Reversibility of 2D cellular automata is undecidable, Physica D 45 (1990) 379–385.
[53] J. Kari, Reversibility and surjectivity problems of cellular automata, J. Comput. System Sci. 48 (1994) 149–182.
[54] J. Kari, Theory of cellular automata: a survey, Theoret. Comput. Sci. 334 (2005) 3–33.
[55] A. Karlsson, Free subgroups of groups with nontrivial Floyd boundary, Commun. Algebra 31 (2003) 5361–5376.
[56] A. Karlsson, G.A. Noskov, Some groups having only elementary actions on metric spaces with hyperbolic boundaries, Geom. Dedicata 104 (2004) 119–137.
[57] A.S. Kechris, Classical descriptive set theory, in: Graduate Texts in Mathematics, vol. 156, Springer-Verlag, New York, 1995.
[58] W. Kuich, On the entropy of context-free languages, Inform. Control 16 (1970) 173–200.
[59] D. Kuske, M. Lohrey, Logical aspects of Cayley-graphs: the monoid case, Int. J. Algebra Comput. 16 (2006) 307–340.
[60] W. Lawton, Note on symbolic transformation groups, Notices of the AMS 19 (1972) A-375. (abstract).
[61] P.A. Linnell, On accessibility of groups, J. Pure Appl. Alg. 30 (1983) 39–46.
[62] R.C. Lyndon, P.E. Schupp, Combinatorial Group Theory, Springer-Verlag Classics in Mathematics, New York, 2000.
[63] A. Machì, F. Mignosi, Garden of Eden configurations for cellular automata on Cayley graphs of groups, SIAM J. Discrete Math. 6 (1993) 44–56.
[64] W. Magnus, Das Identitätsproblem für Gruppen mit einer definerenden relation, Math. Annalen 106 (1932) 295–307.
[65] W. Magnus, A. Karrass, D. Solitar, Combinatorial Group Theory, Wiley, New York, 1966.
[66] M. Margenstern, The Domino Problem of the hyperbolic plane is undecidable, Theoret. Comput. Sci. 407 (2008) 29–84.
[67] M. Margenstern, The Domino Problem of the hyperbolic plane is undecidable, Bull. EATCS 93 (2007) 220–237.
[68] D.A. Martin, Borel determinacy, Ann. Math. (2) 102 (1975) 363–371.
[69] D.A. Martin, A purely inductive proof of Borel determinacy, in: Recursion Theory, in: Proc. Sympos. Pure Math. vol. 42, 1985, Proceedings of the AMSASL summer institute held in Ithaca, New York, 1982, pp. 303–308.
[70] R. McNaughton, Testing and generating infinite sequences by a finite automaton, Inform. Control 9 (1966) 521–530.
[71] S. Moon, A. Valette, Non-properness of amenable actions on graphs with infinitely many ends, In Ischia Group Theory 2006 (2007) 227–233.
[72] E.F. Moore, Machine models of self-reproduction, Proc. Sympos. Appl. Math. 14 (1962) 17–33.
[73] Y.N. Moschovakis, Descriptive set theory, second ed., in: Mathematical Surveys and Monographs, vol. 155, American Mathematical Society, Providence, RI, 2009.
[74] D.E. Muller, Unpublished class notes. University of Illinois at Urbana-Champaign, 1979.
[75] D.E. Muller, Infinite sequences and finite machines, in: Proceedings of the 4th IEEE Symposium on Switching Circuit Theory and Logical Design, 1963, pp. 3–16.
[76] D.E. Muller, P.E. Schupp, Groups, the theory of ends and context-free languages, J. Comput. System Sci. 26 (1983) 295–310.
[77] D.E. Muller, P.E. Schupp, The theory of ends, pushdown automata, and second-order logic, Theoret. Comput. Sci. 37 (1985) 51–75.
[78] D.E. Muller, P.E. Schupp, Context-free languages, groups, the theory of ends, second-order logic, tiling problems, cellular automata, and vector addition systems, Bull. Amer. Math. Soc. 4 (1981) 331–334.
[79] D.E. Muller, P.E. Schupp, Simulating alternating automata by nondeterministic automata: new results and new proofs of the theorems of Rabin, McNaughton and Safra, Theoret. Comput. Sci. 141 (1995) 69–107.
[80] D.E. Muller, P.E. Schupp, Alternating automata on infinite trees, Theoret. Comput. Sci. 54 (1987) 267–276.
[81] J. Myhill, The converse of Moore's garden of Eden theorem, Proc. Amer. Math. Soc. 14 (1963) 685–686.
[82] V.V. Nekrashevych, Self-similar groups, in: Mathematical Surveys and Monographs, vol. 117, Amer. Math. Soc., Providence, RI, 2005.
[83] B.B. Newman, Some results on one-relator groups, Bull. Amer. Math. Soc. 74 (1968) 568–571.
[84] P.S. Novikov, On the algorithmic unsolvability of the word problem in group theory, Trudy Mat. Inst. Steklov 44 (1955) 1–143.
[85] M.O. Rabin, Decidability of second-order theories and automata on infinite trees, Trans. Amer. Math. Soc. 141 (1969) 1–35.
[86] R.M. Robinson, Undecidability and nonperiodicity for tilings of the plane, Invent. Math. 12 (1971) 177–209.
[87] R.M. Robinson, Undecidable tiling problems in the hyperbolic plane, Invent. Math. 44 (1978) 259–264.
[88] P.E. Schupp, A survey of small cancellation theory, in: Boone, Cannonito, Lyndon (Eds.), Word Problems, North-Holland, Amsterdam, 1973.

[89] P.E. Schupp, Groups and graphs: groups acting on trees, ends, and cancellation diagrams, Math. Intell. 1 (1979) 205–222.
[90] G. Senizergues, An effective version of Stallings' theorem in the case of context-free groups, ICALP (1993) 478–495.
[91] G. Senizergues, On the finite subgroups of a context-free group, in: Geometric and Computational Persepectives on Infinite Groups, in: DIMACS Series in Discrete Mathematics and Theoretical Computer Science, vol. 25, American Mathematical Society, 1996.
[92] J. Stallings, On torsion-free groups and infinitely many ends, Ann. of Math. 88 (1968) 312–334.
[93] A. Turing, On computable numbers with an application to the Entscheidungsproblem, Proc. Lond. Math. Soc. 42 (1936) 230–265.
[94] J. von Neumann, in: A.W. Burks (Ed.), The Theory of Self-Reproducing Automata, University of Illinois Press, 1966.
[95] J. von Neumann, Zur Allgemeine Theorie des Masses, Fund. Math. 13 (1929) 73–116.
[96] H. Wang, Proving theorems by pattern recognition II, Bell System Tech. J. 40 (1961) 1–41.
[97] B. Weiss, Sofic groups and dynamical systems. In Ergodic theory and harmonic analysis, Mumbai, 1999, Sankhya Ser. A. 62 (2000) 350–359.
[98] W. Woess, Context-free pairs of groups II — cuts, tree sets, and random walks, Discrete Math. 312 (2012) 157–173.
[99] W. Woess, Fixed sets and free subgroups of groups acting on metric spaces, Math. Z. 214 (1993) 425–439.
[100] W. Woess, Graphs and groups with tree-like properties, J. Combin. Theory, Ser. B 47 (1989) 361–371.
[101] S. Zeitman, Unforgettable forgetful determinacy, J. Logic Comput. 4 (1994) 273–283.